LEARNING CONTEXTUAL INFORMATION FOR HOLISTIC SCENE UNDERSTANDING

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Congcong Li

August 2012

LEARNING CONTEXTUAL INFORMATION FOR HOLISTIC SCENE UNDERSTANDING

Congcong Li, Ph.D.

Cornell University 2012

One of the primary goals in computer vision is holistic scene understanding, which involves many sub-tasks, such as depth estimation, scene categorization, saliency detection, object detection, event categorization, etc. Each of these tasks explains some aspect of a particular scene and in order to fully understand a scene, we would need to solve for each of these sub-tasks. In our human's visual system, the sub-tasks are often coupled together. One task can leverage the output of another task as contextual information for its own decision, and can also feed useful information back to the other tasks. In this thesis, our goal is to design computational algorithms that perform multiple scene understanding tasks in a collaborative way like human does.

In our algorithm design, we consider a two-layer cascade of classifiers, which are repeated instantiations of the original tasks, with the output of the first layer fed into the second layer as input. To better optimize the second-layer outputs, we propose three algorithms, which result in capturing contextual information at multiple levels, ranging from contextual interactions between different tasks to contextual interactions between objects and regions. First, to better leverage the first-layer contextual outputs, we propose Feedback Enabled Cascaded Classification Models (FE-CCM), which jointly optimizes all the sub-tasks. The training of the two-layer cascade involves a feedback step that allows later classifiers to provide earlier classifiers information about which er-

ror modes to focus on, thus results in better combining the contextual information between the various image attributes. Secondly, we also consider sharing contextual information between related tasks. We propose the θ-MRF algorithm, which captures the spatial and semantic relationship between classifiers through an undirected graph built on the parameters. The algorithm encourages the spatially or semantically related classifiers to share their parameters over the contextual features. Third, we discover new contextual attributes from images given only object annotations, to capture the contextual information between objects and regions. We propose two types of visual structured patterns: contextual-meta-object (CMO) and group-of-object (GRP). The CMOs capture multi-scale contextual interactions between objects and the unlabeled regions, and the GRPs capture arbitrary-order interactions between objects that demonstrate consistent spatial, scale, and viewpoint interactions with each other. These contextual patterns are then served as contextual attributes for enhanced object recognition and scene recognition tasks. Finally, we present superior results of the proposed algorithms on a variety of vision applications, including natural scene understanding, images aesthetics assessment, robotic assistive systems, and video applications.

THESIS COMMITTEE

Prof. Tsuhan Chen

Department of Electrical and Computer Engineering,

Cornell University

Prof. Thorsten Joachims

Department of Computer Science,

Cornell University

Prof. Ashutosh Saxena

Department of Computer Science,

Cornell University

Dr. Alexander C. Loui

Kodak Research Laboratories,

Eastman Kodak Company

This thesis is dedicated to my parents:

I couldn't have asked for more.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Prof. Tsuhan Chen, for his great support and guidance through out the five years. Besides the knowledge and experience, I have also been inspired a lot by his optimism towards challenges and his original understandings towards various problems. I also would like to thank him for providing such a flexible environment where I could pursue ideas in multiple sub-areas, and teaching me to think big and think different.

I would like to thank Prof. Ashutosh Saxena, for being a co-advisor on a large part of this thesis. I would like thank him for taking the time and effort to guide and advise me anytime I needed it. It has been a great learning and productive experience working with him.

I also would like to thank Dr. Alexander C. Loui and Prof. Thorsten Joachims for being on my committee and providing the very helpful discussions and suggestions on my thesis. The inputs from the committee members have inspired me a lot in both completing the thesis and pursuing future directions.

I would like to thank my collaborators and senior fellows: Devi Parikh, Andrew Gallagher, Trista Chen, Adarsh Kowdle, Dhruv Batra, David Liu, Wei Yu, Amy Lu, Kate Shim, for their great help and close collaborations. Specifically, I would like to thank Dr. Devi Parikh for always patiently listening to my half-baked ideas, answering my questions, and helping me to find potential research directions. Besides, my work has also greatly benefited from the input of fellow students TP Wong and Norris Xu, ITRI members Chih-Wei Lin, Yi-Ta Wu and Shiaw-Shian Yu, and Microsoft Researcher Dr. Larry Zitnick.

I also would like to thank all AMP members: Kevin Yao-Jen Chang, Zhaoyin Jia, Yimeng Zhang, Amir Sadovnik, Henry Shu, Ruogu Fang, Qi Wu, for sharing our work and life together. The weekly group meetings we had together and the

fun time we enjoyed together are all great memories.

I am grateful for the intellectually stimulating environment at both Cornell and Carnegie Mellon. I have benefited immensely from the seminars and talks and reading groups that I attended. The unending discussions and debates have helped me learn more and think deeper.

In the last, I would like to thank my parents, my husband, and my brother for their full support these years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

One of the key problems in computer vision is holistic scene understanding, which involves understanding many different aspects of an image. When we look at an image of a scene, such as in Figure 1.1, we are often interested in answering several different questions: What objects are there in the image? How far are things? What is going on in the scene? What type of scene is it? How visually appealing is the scene in the viewer's opinion? And so on. These are only a few examples of questions in the area of scene understanding; and there may even be more. Intuitively, the sub-tasks are often coupled. One task can leverage the output of another task as contextual information for its own decision, and can also feed useful information back to the other tasks. For example, recognizing the scene as an indoor scene would provide hints for depth estimation, and vice versa.

Exploiting contextual information between scene understanding tasks has received significant attention in recent works. several models and approaches have exploited such context by combining different classifiers for related tasks in vision[3, 56, 66, 90, 108, 123, 126, 130, 131, 139]; however, most of them tend to be ad-hoc (i.e., a hard-coded rule is used) and often require an intimate knowledge of the inner workings of the individual classifiers and their connections is required. A key challenge here is how to build scalable systems that can simultaneously perform many sub-tasks and intelligently leverage the contextual information between the tasks. In this thesis, our goal is to design learning algorithms that effectively leverage the contextual information between tasks and

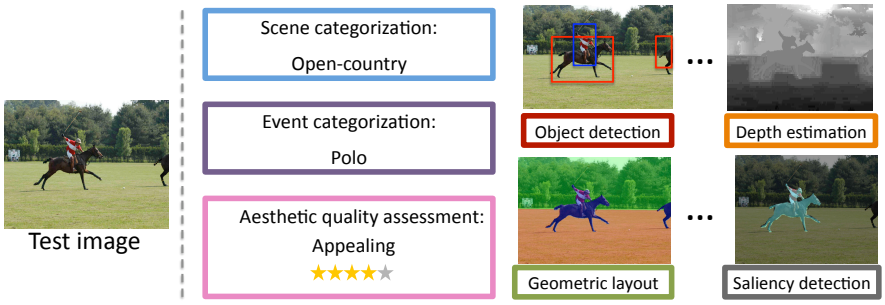


Figure 1.1: Given a test image, Holistic Scene Understanding corresponds to inferring the labels for all possible scene understanding dimensions.

easily scale up for many tasks in large scene understanding systems.

The first question to be answered is what kind of model to be used to connect the large amount of tasks. We want the model to have the following properties: (1) the model simultaneously performs multiple sub-tasks; (2) each task can gain information from the other tasks; (3) the model can automatically discover the strengths of the connections between tasks, and can easily scale up to a large number of tasks. To do this, we investigate a two-layer cascade structure as shown in Figure 1.2. This is also a two-layer instantiation of the cascaded classification models (CCM) proposed by Heitz et al. [53]. Each classifier is repeatedly instantiated on both layers. The first-layer classifiers perform the individual tasks simultaneously and then feed information into the second layer. On the second layer, one task can leverage information from the other tasks by using the output of all the first-layer classifiers as input. We can also easily add new tasks into such a framework. Such a structure has also been explored in some recent research about learning visual attributes towards particular target tasks [29, 69, 145]. Similar to those works, we call the first-layer classifiers "attribute learners" and the second-layer classifiers "target classifiers" in our two-layer structure.

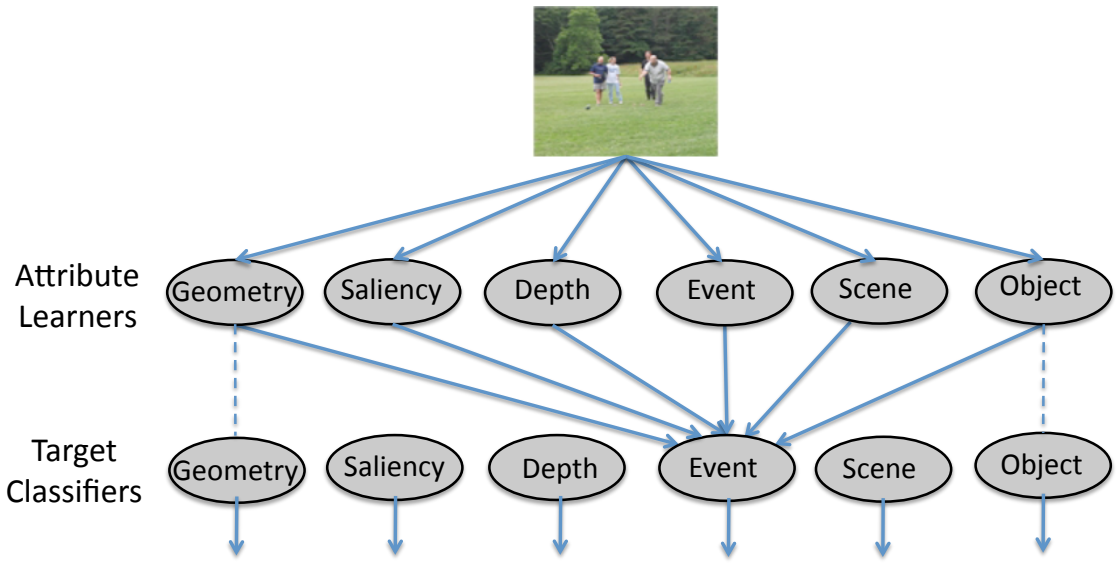


Figure 1.2: The generic cascade structure to be investigated in the thesis.

Based on the two-layer cascade structure in Figure 1.2, the focus of the thesis is to exploit learning algorithms that can better optimize the performance of the target tasks in large scene understanding systems. In previous work by Heitz et al. [53], the classifiers on each layer are trained to the groundtruth labels of corresponding tasks independently. This limits the communications from latter-layer classifiers to earlier ones, and the communications across the different target tasks. In this thesis, in order to better learn the classifiers, we ask the following questions:

- Can we encourage bi-directional communications between the attribute learners and the target classifiers, in order to optimize the final outputs?
- Can we model the interactions between the target classifiers, in order to better learn their parameters especially when the number of classifiers is large?
- Can we introduce new attributes without requiring new types of labels, in order to provide more contextual information for the target tasks?

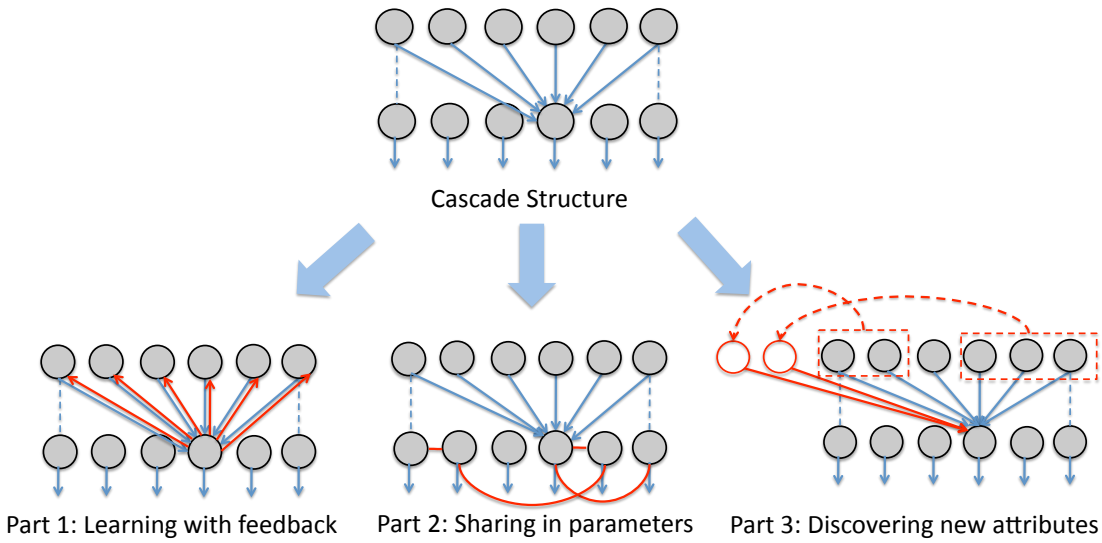


Figure 1.3: An overview of the solutions we propose in this thesis to address the three main questions, based on the generic cascade structure.

In this thesis, we answer these three questions in the three directions as shown in Figure 1.3.

First, we study how to encourage bi-directional communications between the attribute learners and the target classifiers. In the basic structure, we already have information feeding from the first-layer classifiers to the second-layer classifiers. However, there is no feedback from the latter layer to the earlier layer. So we propose introducing feedback from the second-layer classifiers to the first-layer attribute learners during the learning process, as shown in Figure 1.3-Part 1. The motivation of adding feedback is as follows. Some errors made in the first-layer attribute learners are more critical than others. For example, misclassifying a street scene as highway may not hurt as much as misclassifying a street scene as open country. Therefore we prefer the first layer classifier to focus on fixing the latter error instead of optimizing the training accuracy. In another example, allowing the depth estimation to focus on some specific regions can

help perform better scene categorization. For instance, the open country scene is characterized by its upper part as a wide sky area. Thus estimating the depth well in that region by sacrificing some regions in the bottom may help to correctly classify an image. Therefore, we want the feedback to provide earlier stages information about what error modes should be focused on, or what can be ignored without hurting the performance of the later classifiers. We propose a algorithm called Feedback Enabled Cascaded Classification Models (FECCM) in Chapter 2.

Secondly, we investigate the problem of modeling the interactions between the target tasks. This study becomes especially necessary when we consider a large amount of tasks with only limited data for training. The large number of classifiers required for fully understanding a scene results in a large number of parameters to be trained. However, we note that some classifiers are related, spatially or semantically. Given the same inputs (the outputs of the first-layer attribute learners), the parameters for two related classifiers can be similar. To leverage the correlation between the parameters, some previous works enforce priors on the parameters as a directed graph. However, we note that two parameters may not ascribe a directionality to the interaction between them. Therefore, we propose modeling the interactions between parameters through an undirected graph, where the nodes represent the parameters for each specific task and the edges represent the interaction between the parameters, as shown in Figure 1.3-Part 2. We call this representation a θ-MRF, i.e., a Markov Random Field over the parameters, to be described in Chapter 3.

Third, we exploit the problem of discovering new attributes without requiring new types of labels. The existing types of labels may not be informative

enough for the target tasks. For example, the output of "chair" attribute learner and the output of "table" attribute learner is not necessarily helpful for the categorization of a "dining-room" scene. Instead, if we can infer a new attribute like "table set – a table surrounded by four chairs" based on the existing object labels, the output of such an attribute can provide more informative contextual cues for the "dining-room" categorization task on the second layer. Furthermore, we note that the spatial configuration between objects embedded in the new "table set" attribute also results in a more reliable detector than those for individual objects. Therefore, here we consider mining such contextually informative and reliably detectable attributes based on the labels for the existing tasks, as shown in Figure 1.3-Part 3. Specifically, we focus on discovering visual contextual patterns for recognition tasks such as object detection and scene categorization. In Chapter 4 and Chapter 5, we propose discovering two types of visual structured patterns automatically from datasets with only object labels. We then use the discovered patterns to train additional attribute learners, and use their outputs to provide contextual information for the target tasks.

The rest of this thesis is organized as follows. Chapter 2 presents the Feedback Enabled Cascaded Classification Model (FECCM) for combining multiple tasks and leveraging the contextual information between the tasks. Chapter 3 presents our algorithm on building a MRF model over parameters to capture the interactions between the contextually related tasks. Chapter 4 and Chapter 5 presents our efforts on discovering new contextual attributes based on only object annotations used for the existing object attribute learners. We propose discovering two types of structured patterns as mid-level attributes and constructing additional attribute learners for such new attributes. We exploit granularity adaptive contextual regions in Chapter 4 and exploit high-order object

composites in Chapter 5. Chapter 6 then describes applying our proposed algorithms onto various applications: image aesthetics, video analysis, and robotic applications. The thesis is concluded in Chapter 7 with a discussion of potential future work.

1.1 First Published Appearances of Described Contributions

Most contributions or their initial versions described in this thesis have first appeared as various publications:

- Chapter 2: Li, Kowdle, Saxena, Chen [65, 83, 84]
- Chapter 3: Li, Saxena, Chen [76]
- Chapter 4: Li, Parikh, Chen [87]
- Chapter 5: Li, Parikh, Chen [88]
- Chapter 6: Li, Gallagher, Loui, Chen [81], Li, Chen, Dunker, Cremer, Chen [78], Li, Lin, Yu, Chen [85], Li, Wong, Xu, Saxena [89], and Li, Kowdle, Saxena, Chen [83, 84]

The following contributions have appeared as various publications: Li, Chen [79], Li, Loui, Chen [86], Li, Wu, Yu, Chen [82], Yu, Ashraf, Chang, Li, Chen [151], Li, Chen [80]. However, they are beyond the scope of this dissertation, and therefore are not discussed here.

CHAPTER 2

LEARNING WITH FEEDBACK ENABLED CASCADED MODEL

2.1 Introduction

Holistic scene understanding involves many sub-tasks, such as depth estimation, scene categorization, saliency detection, object detection, event categorization, etc. (See Figure 2.1.) Each of these tasks explains some aspect of a particular scene and in order to fully understand a scene, we would need to solve for each of these sub-tasks. Several independent efforts have resulted in good classifiers for each of these sub-tasks. In practice, we see that the sub-tasks are coupled—for example, if we know that the scene is an indoor scene, it would help us estimate depth from that single image more accurately. In another example in the robotic grasping domain, if we know what kind of object we are trying to grasp, then it is easier for a robot to figure out how to pick it up. In this chapter, we propose a unified model that jointly optimizes for all the sub-tasks, allowing them to share information and guide the classifiers towards a joint optimal. We show that this can be seamlessly applied across different applications.

Recently, several approaches have tried to combine these different classifiers for related tasks in vision [3, 56, 66, 90, 108, 123, 126, 130, 131, 139]; however, most of them tend to be ad-hoc (i.e., a hard-coded rule is used) and often an intimate knowledge of the inner workings of the individual classifiers is required. Even beyond vision, in many other domains, state-of-the-art classifiers already exist for many sub-tasks. However, these carefully engineered models are often

tricky to modify, or even to simply re-implement from the available descriptions. Heitz et. al. [53] recently developed a framework for scene understanding called Cascaded Classification Models (**CCM**) treating each classifier as a 'black-box'. Each classifier is repeatedly instantiated with the next layer using the outputs of the previous classifiers as inputs. While this work proposed a method of combining the classifiers in a way that increased the performance in all of the four tasks they considered, it had a drawback that it optimized for each task independently and there was no way of feeding back information from later classifiers to earlier classifiers during training. This feedback can potentially help the CCM achieve a more optimal solution.

In our work, we propose Feedback Enabled Cascaded Classification Models (**FE-CCM**), which provides feedback from the later classifiers to the earlier ones, during the training phase. This feedback, provides earlier stages information about what error modes should be focused on, or what can be ignored without hurting the performance of the later classifiers. For example, misclassifying a street scene as highway may not hurt as much as misclassifying a street scene as open country. Therefore we prefer the first layer classifier to focus on fixing the latter error instead of optimizing the training accuracy. In another example, allowing the depth estimation to focus on some specific regions can help perform better scene categorization. For instance, the open country scene is characterized by its upper part as a wide sky area. Therefore, estimating the depth well in that region by sacrificing some regions in the bottom may help to correctly classify an image. In detail, we do so by jointly optimizing all the tasks; the outputs of the first layers are treated as latent variables and training is done using an iterative algorithm. Another benefit of our method is that each of the classifiers can be trained using their own independent training datasets,

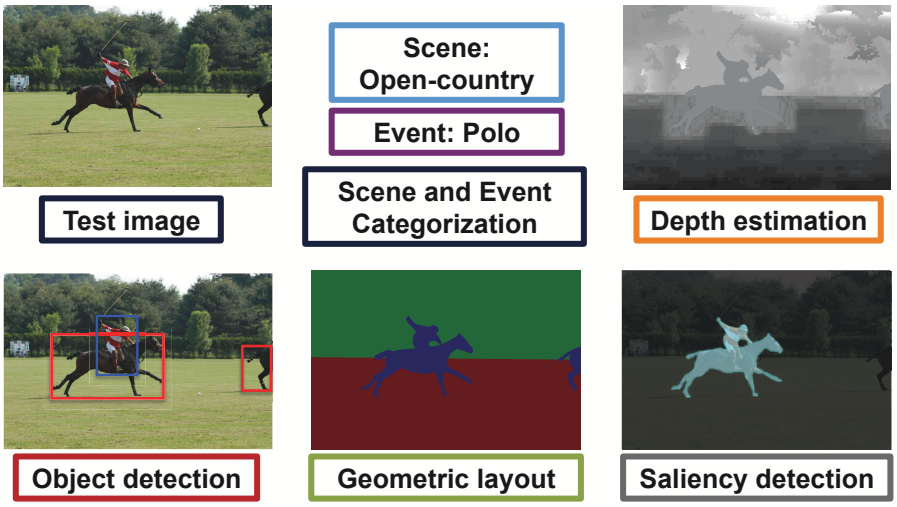


Figure 2.1: Given a test image, *Holistic Scene Understanding* corresponds to inferring the labels for all possible scene understanding dimensions. In our work, we infer labels corresponding to, scene categorization, event categorization, depth estimation (Black = close, white = far), object detection, geometric layout (green = vertical, red = horizontal, blue = vertical) and saliency detection (cyan = salient) as shown above and achieve this jointly using one unified model. Note that different tasks help each other, for example, the depth estimate of the scene can help the object detector look for the horse; the object detection can help perform better saliency detection, etc.

i.e., our model does not require a datapoint to have labels for all the sub-tasks, and hence it scales well with *heterogeneous* datasets.

In our approach, we treat each classifier as a 'black-box', with no restrictions on its operation other than requiring the ability to train on data and have an input/output interface. (Often each of these individual classifier could be quite complex, e.g., producing labelings over pixels in an entire image.) Therefore, our method is applicable to many other tasks that have different but correlated outputs.

In extensive experiments, we show that our method achieves improvements in the performance of *all* the six sub-tasks we consider: depth estimation, object detection, scene categorization, event categorization, geometric labeling and saliency detection. We also successfully apply the same model to robotics and

video applications, to be discussed in Chapter 6.

2.2 Related Work

Cascaded classifiers. Using information from related tasks to improve the performance of the task in question has been studied in various fields of machine learning. The idea of cascading layers of classifiers to aid a task was first introduced with neural networks as multi-level perceptrons where, the output of the first layer of perceptrons is passed on as input to the next layer [16, 39, 50]. However, it is often hard to train neural networks and gain an insight into its operation, making it hard to work for complicated tasks.

The idea of improving classification performance by combining outputs of many classifiers is used in methods such as Boosting [40], where many weak learners are combined to obtain a more accurate classifier; this has been applied to tasks such as face detection [12, 144]. To incorporate contextual information, Fink and Perona [38] exploited local dependencies between objects in a boosting framework, but did not allow for multiple rounds of communication between objects. Torralba et al. [134] introduced Boosted Random Fields to model object dependency, which used boosting to learn the graph structure and local evidence of a conditional random field. Tu [141] proposed a more general framework which used pixel-level label maps to learn a contextual model through a cascaded classifier approach. All these works mainly consider the interactions between labels of the same type.

Recently, Heitz et. al. [53] developed a framework for scene understanding called Cascaded Classification Models (**CCM**), whose focus is on capturing con-

textual interactions between labels of different types. The CCM approach [53] treats each classifier as a 'black-box'. Each classifier is repeatedly instantiated with the next layer using the outputs of the previous classifiers as inputs. However, it had a drawback that it optimized for each task independently and there was no way of feeding back information from later classifiers to earlier classifiers during training. This feedback can potentially help the CCM achieve a more optimal solution. Therefore, in our work, we proposed a method that allows feeding back information from the latter layer to the earlier layer, and optimize the target outputs jointly. Besides, compared to the cascade method in [141], our model with feedback not only iteratively refines the contextual interactions, but also refines the individual classifiers to provide helpful context.

Sensor fusion. There has been a huge body of work in the area of sensor fusion where classifiers output the same labels but work with different modalities, each one giving additional information and thus improving the performance, e.g., in biometrics, data from voice recognition and face recognition is combined [63]. However, in our scenario, we consider multiple tasks where each classifier is tackling a different problem (i.e., predicting different labels), with the same input being provided to all the classifiers.

Structured Models for combining tasks. While the methods discussed above combine classifiers to predict the *same* labels, there is a group of works that designs models for predicting heterogenous labels. Kumar and Hebert [66] developed a large MRF-based probabilistic model to link multi-class segmentation and object detection. Li et al. [77] modeled mulitple interactions within tasks and across tasks by defining a MRF over parameters. Similar efforts have been made in the field of natural language processing. Sutton and McCallum

[131] combined a parsing model with a semantic role labeling model into a unified probabilistic framework that solved both simultaneously. Ando and Zhang [4] proposed a general framework for learning predictive functional structures from multiple tasks. All these models require knowledge of the inner workings of the individual classifiers, which makes it hard to fit existing state-of-the-art classifiers of certain tasks into the models.

Structured learning algorithms (e.g., [64, 132, 140]) can also be a viable option for the setting of combining multiple tasks. There has been recent development in structured learning on handling latent variables (e.g. hidden conditional random field [112], latent structured SVM [150]), which can be potentially applied to multi-task settings with disjoint datasets. With considerable understanding into each of the tasks, the loss function in structured learning provides a nice way to leverage different tasks. However, in this work, we focus on developing a more generic algorithm that can be easily applied even without intimate knowledge of the tasks.

There have been many works which show that with a well-designed model, one can improve the performance of a particular task by using cues from other tasks (e.g., [3, 108, 139]). Saxena et al. manually designed the terms in an MRF to combine depth estimation with object detection [123] and stereo cues [126]. Sudderth et al. [130] used object recognition to help 3D structure estimation.

Context. There is a large body of work that leverages contextual information to help specific tasks. Various sources of context have been explored, ranging from the global scene layout, interactions between objects and regions to local features. To incorporate scene-level information, Torralba et al. [137, 138] used the statistics of low-level features across the entire scene to prime object detection or

help depth estimation. Hoiem et al. [57] used 3D scene information to provide priors on potential object locations. Park et al. [110] used the ground plane estimation as contextual information for pedestrian detection. Many works also model context to capture the local interactions between neighboring regions [54, 67, 94], objects [22, 33, 108, 113, 148], or both [7, 24, 41]. These methods improve the performance of some specific tasks by combining information from different aspects. However, most of these methods can not be applied to cases when we only have "black-box" classifiers for the individual tasks.

Holistic Scene Understanding. Hoiem et. al. [56] proposed an innovative but ad-hoc system that combined boundary detection and surface labeling by sharing some low-level information between the classifiers. Li et. al. [90, 92] combined image classification, annotation and segmentation with a hierarchical graphical model. However, these methods required considerable attention to each classifier, and considerable insight into the inner workings of each task and also the connections between them. This limits the generality of the approaches in introducing new tasks easily or being applied to other domains.

Deep Learning. There is also a large body of work in the areas of deep learning, and we refer the reader to Bengio and LeCun [6] for a nice overview of deep learning architectures and Caruana [13] for multitask learning with shared representation. While efficient back-propagation methods like [72] have been commonly used in learning a multi-layer network, it is not as easy to apply to our case where each node is a complex classifier. Most works in deep learning (e.g., [46, 55, 152]) are different from our work in that, those works focus on one particular task (same labels) by building different classifier architectures, as compared to our setting of *different* tasks with different labels. Hinton et al. [55]

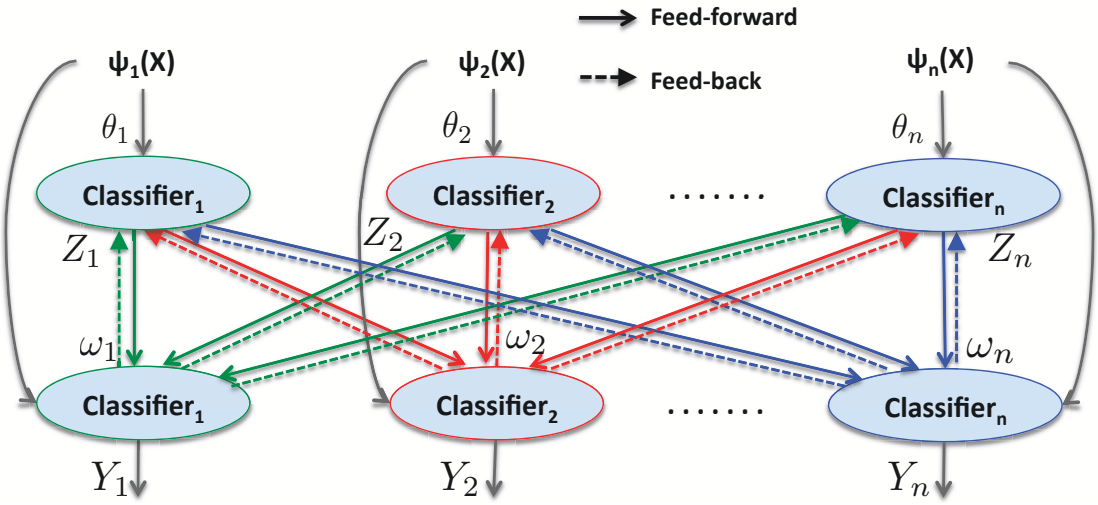


Figure 2.2: The proposed feed-back enabled cascaded classification model (FE-CCM) for combining related classifiers. ($\forall i \in \{1, 2, \ldots, n\}$, $\Psi_i(X)$ = Features corresponding to $Classifier_i$ extracted from image X, Z_i = Output of the $Classifier_i$ in the first stage parameterized by θ_i, Y_i = Output of the $Classifier_i$ in the second stage parameterized by ω_i). In the proposed FE-CCM model, there is feed-back from the latter stages to help achieve a model which optimizes all the tasks considered, jointly. Here $Classifier_i$'s on the two layers can have different forms though they are for the same classification task. (Note that different colors of lines are used only to make the figure more readable.)

used unsupervised learning to obtain an initial configuration of the parameters. This provides a good initialization and hence their multi-layered architecture does not suffer from local minimas during optimization. At a high-level, we can also look at our work as a multi-layered architecture (where each node typically produces complex outputs, e.g., labels over the pixels in the image); and initialization in our case comes from existing state-of-the-art individual classifiers. Given this initialization, our training procedure finds parameters that (consistently) improve performance across all the sub-tasks.

2.3 Approach: FE-CCM

In the field of scene understanding, a lot of independent research into each of the vision sub-tasks has led to excellent classifiers. These independent classifiers are

typically trained on different or *heterogenous* datasets due to the lack of ground-truth labels for all the sub-tasks. In addition, each of these classifiers come with their own learning and inference methods. Usually the goal of each task is to produce a label $Y_i \in \mathbf{S_i}$ for the i^{th} sub-task. If we are considering depth estimation (see Figure 2.1), then the label would be $Y_1 \in \mathbf{S_1} = \mathbb{R}_+^{100\times100}$ for continuous values of depth in a 100×100 output. For scene categorization, we will have $Y_2 \in \mathbf{S_2} = \{1, \ldots, K\}$ for K scene classes. If we have n sub-tasks, then we would have to produce an output as:

$$\mathcal{Y} = \{Y_1, \ldots, Y_n\} \in \mathbf{S_1} \times \mathbf{S_2} \ldots \times \mathbf{S_n}.$$

The interesting part here is that often we want to solve different combinations of the sub-tasks depending on the situation. Our goal is to design an algorithm that does not depend on the particular sub-tasks in question. Therefore, we want to consider each of them as a 'black-box', which makes it easy to combine them. We describe what we mean by 'black-box classifiers' below.

Black-box Classifier. A black-box classifier, as the name suggests, is a classifier for which operations (such as learning and inference algorithms) are available for use, but their inner workings are not known. We assume that, given the training dataset $\mathbf{X}$, features extracted $\mathbf{\Psi}(\mathbf{X})$ and the target outputs of the i^{th} task $\mathbf{Y_i}$, the black-box classifier has some internal learning function f^i_{learn} with parameters θ_i that optimizes the mapping from the inputs to the outputs for the training data. [1] Once the parameters have been learnt, given a new data point, X with features $\Psi(X) \in \mathbb{R}^K$, where K can be changed as desired,[2] the black-box classifier returns the output $\hat{Y}_i$ according to its internal inference function f^i_{infer}.

[1] Unless specified, the regular symbols (e.g. X, Y_i, etc.) are used for a particular data-point, and the bold-face symbols (e.g. $\mathbf{X}$, $\mathbf{Y_i}$, etc.) are used for a dataset.

[2] If the input dimension of the black-box classifier can not be changed, then we will use that black-box in the first layer only.

This is illustrated through the equations below. For the i^{th} task,

$$Learning : \theta_i^* = \underset{\theta_i}{\text{optimize}}\, f^i_{\text{learn}}(\mathbf{\Psi}(\mathbf{X}), \mathbf{Y_i}; \theta_i) \tag{2.1}$$

$$Inference : \hat{Y}_i = \underset{Y_i}{\text{optimize}}\, f^i_{\text{infer}}(\Psi(X), Y_i; \theta_i^*). \tag{2.2}$$

This approach of treating each classifier as a black-box allows us to use different existing classifiers which have been known to perform well at specific sub-tasks. Furthermore, without changing their inner workings, it allows us to compose them into one model which exploits the information from each sub-task to aid holistic scene understanding.

2.3.1 Feedback Enabled Cascaded Classification Models

Our model is built in the form of a two-layer cascade, as shown in Figure 2.2. The first layer consists of an instantiation of each of the black-box classifiers with the image features as input. The second layer is a repeated instantiation of each of the classifiers with the first layer classifier outputs as well as the image features as inputs. Note that the repeated classifier on the second layer is not necessary to have the same mathematical form with the one on the first layer. Instead, we consider it as a repeated instantiation only because they are used for the same classification task.

Notation: We consider n related sub-tasks Classifier$_i$, $i \in \{1, 2, \ldots, n\}$ (Figure 2.2). We describe the notations used in this chapter as follows:

With the notations in place we will now first describe the inference and learning algorithms for the proposed model in the following sections, followed by probabilistic interpretation of our method.

$\Psi_i(X)$	Features corresponding to Classifier$_i$ extracted from image X.
Z_i, $\mathcal{Z}$	Z_i indicates output from the first layer Classifier$_i$. Many classifiers output continuous scores instead of labels. In cases where this is not the case, it is trivial to convert a binary classiers output to a log-odds scores. For a K-class ($K > 2$) classifier, we consider the output to be a K-dimensional vector. $\mathcal{Z}$ indicates the set $\{Z_1, Z_2, \ldots, Z_n\}$.
θ_i, Θ	θ_i indicates parameters corresponding to first layer Classifier$_i$. Θ indicates the set $\{\theta_1, \ldots, \theta_n\}$.
Y_j, $\mathcal{Y}$	Y_j indicates output for the j^{th} task in the second layer, using the original features $\Psi_j(X)$ as well as all the outputs from the first layer as input. $\mathcal{Y}$ indicates the set $\{Y_1, Y_2, \ldots, Y_n\}$.
ω_j, Ω	ω_j indicates parameters for the second layer Classifier$_j$. Ω indicates the set $\{\omega_1, \ldots, \omega_n\}$.
Γ_j, Γ	Dataset for the j^{th} task, which consists of labeled pairs $\{X, Y_j\}$ in the training set. Γ represents all the labeled data.
f^i_{infer}, f^i_{learn}	the internal inference function and learning function for the i^{th} classifier on the first layer.
f'^i_{infer}, f'^i_{learn}	the internal inference function and learning function for the i^{th} classifier on the second layer.

2.3.2 Inference Algorithm

During inference, the inputs $\Psi_i(X)$ are given and our goal is to infer the final outputs Y_i. Using the learned parameters θ_i for the first level of classifiers and ω_i for the second level of classifiers, we first infer the first-layer outputs Z_i and then infer the second-layer outputs Y_i. More formally, we perform the following.

$$\hat{Z}_i = \underset{Z_i}{\text{optimize}}\, f^i_{\text{infer}}(\Psi_i(X), Z_i; \hat{\theta}_i) \tag{2.3}$$

$$\hat{Y}_i = \underset{Y_i}{\text{optimize}}\, f'^i_{\text{infer}}([\Psi_i(X)\ \hat{\mathcal{Z}}], Y_i; \hat{\omega}_i) \tag{2.4}$$

The inference algorithm is given in Algorithm 1. This method allows us to use the internal inference function (Equation 2.2) of the black-box classifiers without knowing its inner workings. Note that the complexity here is no more than constant times the complexity of inference in the original classiers.

Algorithm 1: Inference

1. **Inference for first layer:**

   for $i = 1 : n$

      Infer the outputs of the i^{th} classifier using Equation 2.3;

   end

2. **Inference for second layer:**

   for $i = 1 : n$

      Infer the outputs of the i^{th} classifier using Equation 2.4;

   end

2.3.3 Learning Algorithm

During the training stage, the inputs $\Psi_i(X)$ as well as the target outputs, $Y_1, Y_2, \ldots, Y_n$ of the second level of classifiers, are all observed (because the ground-truth labels are available). In our algorithm, we consider $\mathcal{Z}$ (outputs of layer 1 and inputs to layer 2) as hidden variables. In previous work, Heitz et al. [53] assume that each layer is independent and that each layer produces the best output independently (without consideration for other layers), and therefore use the ground-truth labels for $\mathcal{Z}$ even for training the classifiers in the first layer.

On the other hand, we want to optimize for the final outputs as much as possible. Thus the first layer classifiers need not perform their best (w.r.t. groundtruth), but rather focus on error modes that would result in the second layer's output $(Y_1, Y_2, \ldots, Y_n)$ being more correct. Therefore, we learn the model through an iterative Expectation-Maximization formulation, given the indepen-

dencies between classifiers represented by the model in Figure 2.2. In one step (Feed-forward step) we assume the variables Z_i's are known and learn the parameters and in the other step (Feed-back step) we fix the parameters estimated previously and estimate the variables Z_i's. Since the Z_i's are not fixed to the ground truth, as the iterations progress, the first level of classifiers start focusing on the error modes which would give the best improvement in performance at the end of the second level of classifiers. The learning algorithm is summarized in Algorithm 2.

Algorithm 2: Learning

1. Initialize latent variables $\mathcal{Z}$ with the ground-truth $\mathcal{Y}$.

2. Do until convergence or maximum iteration: {

Feed-foward step: Fix latent variables $\mathcal{Z}$, estimate the parameters Θ and Ω using Equation 2.5 and Equation 2.6.

Feed-back step: Fix the parameters Θ and Ω, compute latent variables $\mathcal{Z}$ using Equation 2.7.

}

Initialization: We initialize the model by setting the latent variables Z_i's to the groundtruth, i.e. $Z_i = Z_i^{gt}$. Training with this initialization, our cascade is equivalent to CCM in [53], where the classifiers (and the parameters) in the first layer are similar to the original state-of-the-art classifier and the classifiers in the second layer use the outputs of the first layer in addition to the original features as input.

Feed-forward Step: In this step, we estimate the parameters Θ and Ω. We assume that the latent variables Z_i's are known (and Y_i's are known because they are the ground-truth during learning, i.e. $Y_i = Y_i^{gt}$). We then learn the parameters

of each classifier independently. Learning θ_i is precisely the learning problem of the 'black-box classifier', and learning ω_i is also an instantiation of the original learning problem, but with the original input features appended with the outputs of the first level classifiers. Therefore, we can use the learning method provided by the individual black-box classifier (Equation 2.1).

$$\hat{\theta}_i = \underset{\theta_i}{\text{optimize}}\ f^i_{\text{learn}}(\mathbf{\Psi_i}(\mathbf{X}), \mathbf{Z}_i; \theta_i) \tag{2.5}$$

$$\hat{\omega}_i = \underset{\omega_i}{\text{optimize}}\ f'^i_{\text{learn}}([\mathbf{\Psi_i}(\mathbf{X})\ \mathcal{Z}], \mathbf{Y}_i; \omega_i) \tag{2.6}$$

We now have the parameters for all the classifiers.

Feed-back Step: In the second step, we will estimate the values of the variables Z_i's assuming that the parameters are fixed (and Y_i's are given because the ground-truth is available, i.e. $Y_i = Y_i^{gt}$). This feed-back step is the crux that provides information to the first-layer classifiers what error modes should be focused on and what can be ignored without hurting the final performance. Given θ_i's and ω_i's are fixed, we want the Z_i's to be good predictions from the first-layer classifiers and also help to increase the correction predictions of Y_i's as much as possible. We optimize the following function for the feed-back step:

$$\underset{\mathcal{Z}}{\text{optimize}} \sum_{i=1}^{n} \left(J^i_1(\Psi_i(X), Z_i; \hat{\theta}_i) + J^i_2(\Psi_i(X), \mathcal{Z}, Y_i; \hat{\omega}_i) \right) \tag{2.7}$$

where J^i_1's and J^i_2's are functions respectively related to the first-layer classifiers and the second-layer classifiers. one option is to have $J^i_1(\Psi_i(X), Z_i; \hat{\theta}_i) = f^i_{\text{infer}}(\Psi_i(X), Z_i; \hat{\theta}_i)$ and $J^i_2(\Psi_i(X), \mathcal{Z}, Y_i; \hat{\omega}_i) = f'^i_{\text{infer}}([\Psi_i(X), \mathcal{Z}], Y_i; \hat{\omega}_i)$ if the intrinsic inference functions for the classifiers are known. More discussions will be given in Section 2.3.4 if the intrinsic functions are unknown. The updated Z_i's will be

used to re-learn the classifier parameters in the feed-forward step of next iteration. Note that the updated Z_i's have continuous values. If the internal learning function of a classifier accepts only labels, we threshold the values of Z_i's to get labels.

2.3.4 Probabilistic Interpretation

Our algorithm can be explained with a probabilistic interpretation where the goal is to maximize the log-likelihood of the outputs of all tasks given the observed inputs, i.e., $\log P(\mathcal{Y}|X)$, where X is an image belonging to training set Γ. Therefore, the goal of the proposed model shown in Figure 2.2 is to maximize

$$\log \prod_{X \in \Gamma} P(\mathcal{Y}|X; \Theta, \Omega) \tag{2.8}$$

To introduce the hidden valuables Z_i's, we expand Equation 2.8 as follows, using the independencies represented by the directed model in Figure 2.2.

$$= \sum_{X \in \Gamma} \log \sum_{\mathcal{Z}} P(Y_1, \ldots, Y_n, \mathcal{Z}|X; \Theta, \Omega) \tag{2.9}$$

$$= \sum_{X \in \Gamma} \log \sum_{\mathcal{Z}} \prod_{i=1}^{n} P(Y_i|\Psi_i(X), \mathcal{Z}; \omega_i) P(Z_i|\Psi_i(X); \theta_i) \tag{2.10}$$

However, the summation inside the log makes it difficult to learn the parameters. Motivated by the Expectation Maximization algorithm [21], we iterate between the two steps as described in the following. Again we initialize the classifiers by learning the classifiers with ground-truth as discussed Section 2.3.3.

Feed-forward Step: In this step, we estimate the parameters by assuming that the latent variables Z_i's are known (and Y_i's are known anyway because they are the ground-truth). This results in

$$\underset{\theta_1,\ldots,\theta_n,\omega_1,\ldots,\omega_n}{\text{maximize}} \sum_{X\in\Gamma} \log \prod_{i=1}^{n} P(Y_i|\Psi_i(X), \mathcal{Z}; \omega_i) P(Z_i|\Psi_i(X); \theta_i) \tag{2.11}$$

Now in this feed-forward step, the terms for maximizing the different parameters turn out to be independent. So, for the i^{th} classifier we have:

$$\underset{\theta_i}{\text{maximize}} \sum_{X\in\Gamma} \log P(Z_i|\Psi_i(X); \theta_i) \tag{2.12}$$

$$\underset{\omega_i}{\text{maximize}} \sum_{X\in\Gamma} \log P(Y_i|\Psi_i(X), \mathcal{Z}; \omega_i) \tag{2.13}$$

Note that the optimization problem nicely breaks down into the sub-problems of training the individual classifier for the respective sub-tasks. We can solve each sub-problem separately given the probabilistic interpretation of the corresponding classifier. When the classifier is taken as 'black-box', this can be approximated using the original learning method provided by the individual black-box classifier (Equation 2.5 and Equation 2.6)

Feed-back Step: In this step, we estimate the values of the latent variables Z_i's assuming that the parameters are fixed. We perform MAP inference on Z_i's (and not marginalization). This can be considered as a special variant of the general EM framework (hard EM, [103]). Using Equation 2.10, we get the following optimization problem:

$$\begin{aligned} &\underset{\mathcal{Z}}{\text{maximize}} \log P(Y_1, \ldots, Y_n, \mathcal{Z}|X; \hat{\theta}_1, \ldots, \hat{\theta}_n, \hat{\omega}_1, \ldots, \hat{\omega}_n) \Leftrightarrow \\ &\underset{\mathcal{Z}}{\text{maximize}} \sum_{i=1}^{n} \Big(\log P(Y_i|\Psi_i(X), \mathcal{Z}; \hat{\omega}_i) + \log P(Z_i|\Psi_i(X); \hat{\theta}_i) \Big) \end{aligned} \tag{2.14}$$

This maximization problem requires that we have access to the characterization of the individual black-box classifiers in a probabilistic form. If the probabilistic interpretations of the classifiers are known, we can solve the above

function accordingly. Note that Equation 2.14 is same as Equation 2.7 with $J_1^i(\Psi_i(X), Z_i; \hat{\theta}_i) = \log P(Z_i|\Psi_i(X); \hat{\theta}_i)$ and $J_2^i(\Psi_i(X), \mathcal{Z}, Y_i; \hat{\omega}_i) = \log P(Y_i|\Psi_i(X), \mathcal{Z}; \hat{\omega}_i)$.

In some cases, the classifier log-likelihoods in Equation 2.14 actually turn out to be convex. For example, if the individual classifiers are linear or logistic classifiers, the minimization problem is convex and can be solved using gradient descent (or any such method).

However, if the probabilistic interpretations of the classifiers are unknown, the feedback step requires extra modeling. Some modeling options are provided as follows:

- Case 1: Insight into the vision problem is available. In this case, one could use the domain knowledge of the task into the problem to properly model J_1^i's and J_2^i's.
- Case 2: No insight into the vision problem is available and no internal function of the original classifier is known. In this case, we formulate the J_1^i's and J_2^i's as follows. The J_1^i is defined to be a distance function between the target Z_i and the estimated $\hat{Z}_i$, which serves as a regularization for the first-layer classifiers.

$$\begin{aligned} J_1^i(\Psi_i(X), Z_i; \hat{\theta}_i) &= \left\| Z_i - \hat{Z}_i \right\|^2 \\ \text{s.t. } \hat{Z}_i &= \underset{Z_i}{\text{optimize}}\, f_{\text{infer}}^i(\Psi_i(X), Z_i; \hat{\theta}_i) \end{aligned} \tag{2.15}$$

  To formulate J_2^i's, we make a variational approximation on the output of the second-layer classifier for task i (i.e., approximating it as a Gaussian, [45]) to get:

$$\underset{\alpha_i}{\text{minimize}} \sum_{X \in \Gamma} \left\| \hat{Y}_i - \alpha_i^T [\Psi_i(X), \hat{\mathcal{Z}}] \right\|_2^2 \tag{2.16}$$

  where α_i are parameters of the approximation model. $\hat{Y}_i$ is the actual output of the second layer classifier for the task i, i.e. $\hat{Y}_i$ =

optimize$_{Y_i}$ $f'_{\text{infer}}([\Psi_i(X)\ \hat{\mathcal{Z}}], Y_i; \hat{\omega}_i)$. Then we define the J_2^i's as follows.

$$J_2^i(\Psi_i(X), \mathcal{Z}, Y_i; \hat{\omega}_i) = \left\| Y_i - \hat{\alpha}_i^T [\Psi_i(X), \mathcal{Z}] \right\|_2^2 \tag{2.17}$$

Sparsity: Note that the parameter α_i is typically extremely high-dimensional (and increases with the number of tasks) because the second layer classifiers take as input the original features as well as outputs of all previous layers. The learning for the approximation model may become ill-conditioned. Therefore, we want our model to select only a few non-zero weights, i.e., only a few non-zero entries in α_i. We do this by introducing the l_1 sparsity in the parameters [96]. So Equation 2.16 is extended as follows.

$$\underset{\alpha_i}{\text{minimize}} \sum_{X \in \Gamma} \left(\left\| \hat{Y}_i - \alpha_i^T [\Psi_i(X), \hat{\mathcal{Z}}] \right\|_2^2 + \beta |\alpha_i| \right) \tag{2.18}$$

Inference: As introduced in Section 2.3.2, our inference procedure consists of two steps: first maximize over hidden variable Z and then maximize over Y. [3]

$$\hat{\mathcal{Z}} = \underset{\mathcal{Z}}{\text{argmax}} \log P(\mathcal{Z}|X, \hat{\Theta}) \tag{2.19}$$

$$\hat{\mathcal{Y}} = \underset{\mathcal{Y}}{\text{argmax}} \log P(\mathcal{Y}|\hat{\mathcal{Z}}, X, \hat{\Omega}) \tag{2.20}$$

Given the structure of our directed graph, the outputs for different classifiers on the same layer are independent given their inputs and parameters. Therefore, Equations 2.19 and 2.20 are equivalent to the following:

$$\hat{Z}_i = \underset{Z_i}{\text{argmax}} \log P(Z_i|\Psi_i(X); \hat{\theta}_i), i = 1, \ldots, n \tag{2.21}$$

$$\hat{Y}_i = \underset{Y_i}{\text{argmax}} \log P(Y_i|\Psi_i(X); \hat{\mathcal{Z}}; \hat{\omega}_i), i = 1, \ldots, n \tag{2.22}$$

As we see, Equation 2.21 and Equation 2.22 are instantiations of Equation 2.3 and Equation 2.4 in the probabilistic form.

[3] Another alternative would have been to maximize $P(Y|X) = \sum_Z P(Y, Z|X)$; however, this would require marginalization over the variable Z which is expensive to compute.

2.3.5 Training with Heterogeneous datasets

Often real datasets are disjoint for different tasks, i.e, each datapoint does not have the labels for all the tasks. Our formulation handles this scenario well. In this section, we show our formulation for this general case, where we use Γ_i as the dataset that has labels only for the i^{th} task.

In the following we provide the modifications to the feed-forward step and the feed-back step while dealing with disjoint datasets, i.e., data in dataset Γ_i only have labels for the i^{th} task. These modifications also allow us to develop different variants of the model, described later in this section.

Feed-forward Step: Using the feedback step, we can have Z_i's for all the data. Therefore, we use all the datasets in order to re-learn each of the first-layer classifiers. If the internal learning function of the black-box classifier is additive over the data points, then we have

$$\hat{\theta}_i = \underset{\theta_i}{\text{optimize}} \sum_j \sum_{X \in \Gamma_j} \pi_j f^i_{\text{learn}}(\Psi_i(X), Z_i; \theta_i), \tag{2.23}$$

where π_j's are the importance factors given to different datasets, and satisfy $\sum_j \pi_j = 1$. (See the later part of this section on how to choose π_j's.)

If the internal learning function is not additive over the data points, we provide an alternative solution here. We sample a subset of data $\mathbf{X}^j$ from each dataset Γ^j, i.e. $\mathbf{X}^j \subseteq \Gamma^j$ and combine them into a new set $\mathbf{X} = [\mathbf{X}^1, \ldots, \mathbf{X}^n]$. In $\mathbf{X}$, the ratio of data belonging to $\mathbf{X}^j$ is equal to π_j, i.e. $\frac{|\mathbf{X}^j|}{|\mathbf{X}|} = \pi_j$, where $|\cdot|$ indicates the number of data-points. Then we can learn the parameters of the first-layer classifiers as follows.

$$\hat{\theta}_i = \underset{\theta_i}{\text{optimize}}\ f^i_{\text{learn}}(\mathbf{\Psi_i}(\mathbf{X}), \mathbf{Z_i}; \theta_i), \tag{2.24}$$

To re-learn the second-layer classifiers, the only change made to Equation 2.6 is that instead of using all data while optimizing for a particular task, we use only the data-points that have the ground-truth label for the corresponding task.

$$\hat{\omega}_i = \underset{\omega_i}{\text{optimize}}\ f'^{i}_{\text{learn}}([\mathbf{\Psi_i}(\mathbf{X})\ \mathcal{Z}], \mathbf{Y}_i; \omega_i),\ \text{s.t. } \mathbf{X} = \Gamma_i \tag{2.25}$$

Feed-back Step: In this step, we change Equation 2.7 as follows. Since a datapoint in the set Γ_j only has ground-truth label for the j^{th} task (Y_j), we only consider J_2^j in the second term. However, since this datapoint has outputs for all the first-layer classifiers using the feed-forward step, we consider all the J_1^i's, $i = 1, \cdots, n$. Therefore, in order to obtain the value of $\mathcal{Z}$ corresponding to each data-point $X \in \Gamma_j$, we have

$$\underset{\mathcal{Z}}{\text{optimize}} \sum_{i=1}^{n} (J_1^i(\Psi_j(X), Z_i; \hat{\theta}_i)) + J_2^j(\Psi_j(X), \mathcal{Z}, Y_j; \hat{\omega}_j). \tag{2.26}$$

The parameters π_j allow us to formulate three different instantiations of our model.

- **Unified FECCM**: In this instantiation, our goal is to achieve improvements in all tasks with one set of parameters $\{\Theta, \Omega\}$. We want to balance the data from different datasets (i.e., with different task labels). Towards this goal, π_j is set to be inversely proportional to the amount of data in the dataset of the j^{th} task. Therefore, the unified FECCM balances the amount of data in different datasets, based on Equation 2.23.

- **One-goal FECCM**: In this instantiation, we set $\pi_j = 1$ if $j = k$, and $\pi_j = 0$ otherwise. This is an extreme setting to favor the specific task k. In this case, the retraining of the first-layer classifiers will only use the feedback from the Classifier$_k$ on the second layer, i.e., only use the dataset with labels for the k^{th}

task. Therefore, FECCM degrades to a model with only one target task (the k^{th} task) on the second layer and all the other tasks are only instantiated on the first layer. Although the goal in this setting is to completely benefit the k^{th} task, in practice it often results in overfitting and does not always achieve the best results even for the specific task (see Table 2.1 in Section 2.5). In this case, we train different models, i.e. different θ_i's and ω_i's, for different target tasks.

- **Target-Specific FECCM**: This instantiation is to optimize the performance of a specific task. As compared to one-goal FECCM where we manually remove the other tasks on the second layer, in this instantiation we keep all the tasks on the second layer and conduct data-driven selection of the parameters π_j for different datasets. In detail, π_j is selected through cross validation on a hold-out set in the learning process in order to optimize the second-layer output of a specific task. Since Target-Specific FECCM still has all the tasks instantiated on the second layer, the re-training of the first-layer classifiers can still use data from different datasets (i.e., with different task labels). Here we train different models, i.e. different θ_i's and ω_i's, for different target tasks.

2.3.6 Computational Efficiency

This method allows us to use the internal training and inference function (Equation 2.1 and Equation 2.2) of the black-box classifiers without knowing its inner workings. In the training stage, the learning algorithm iterates between the feed-forward step and the feed-back step. In the feed-forward step, the complexity is no more than constant times the complexity of inference in the original classiers. Note that the inferences for the classifiers on the same layer can be easily paralleled. In the feed-back step, as we approximate the second-layer

classifiers with gaussian models, the feedback process only requires solving a quadratic problem and adds little overhead to the computation of each iteration. In our experiments on six scene understanding tasks, the training contains 4-5 iterations which takes about 5-6 hours on a Macintosh machine with 2.66GHz CPU. In the testing stage, our algorithm only requires a single run of the feed-forward inference process, the complexity of which is no more than constant times the complexity of inference in the original classiers.

2.4 Implementation

In this section we describe the implementation details of our instantiation of FE-CCM for scene understanding. Each of the classifiers described below for the sub-tasks are our "base-model" shown in Table 2.1. In some sub-tasks, our base-model will be simpler than the state-of-the-art models (that are often hand-tuned for the specific sub-tasks respectively). However, even when using base-models in our FE-CCM, our model will still outperform the state-of-the-art models for the respective sub-tasks (on the same standard respective datasets) in Section 2.5.

In order to explain the implementation details for the different tasks, we will use the following notation. Let i be the index of the tasks we consider. We consider 6 tasks for our experiments on scene understanding: scene categorization ($i = 1$), depth estimation ($i = 2$), event categorization ($i = 3$), saliency detection ($i = 4$), object detection ($i = 5$) and geometric labeling ($i = 6$). The inputs for the j^{th} task at the first layer are given by the low-level features Ψ_j. At the second layer, in addition to the original features Ψ_j, the inputs include the outputs from

the first layer classifiers. This is given by,

$$\Phi_j = \left[\Psi_j \, Z_1 \, Z_2 \, Z_3 \, Z_4 \, Z_5 \, Z_6\right] \tag{2.27}$$

where, Φ_j is the input feature vector for the j^{th} task on the second layer, and Z_i ($i = 1, \ldots, 6$) represents the output from the i^{th} task which is appended to the input to the j^{th} task on the second layer and so on.

Scene Categorization. For scene categorization, we classify an image into one of the 8 categories defined by Torralba et al. [105] tall building, inside city, street, highway, coast, open country, mountain and forest. We evaluate the performance by measuring the rate of incorrectly assigning a scene label to an image on the MIT outdoor scene dataset [105]. The feature inputs for the first-layer scene classifier $\Psi_1 \in \mathbb{R}^{512}$ is the GIST feature [105], extracted at 4×4 regions of the image, on 4 scales and 8 orientations.

We use an RBF-Kernel SVM classifier [135], as the first-layer scene classifier, and a multi-class logistic classifier for the second layer. The output of the first-layer scene classifier $Z_1 \in \mathbb{R}^8$ is an 8-dimensional vector where each element represents the log-odds score of the corresponding image belonging to a scene category. This 8-dimensional output is fed to each of the second-layer classifiers.

Depth Estimation. For the single image depth estimation task, we estimate the depth of every pixel in an image. We evaluate the estimation performance by computing the root mean square error of the estimated depth with respect to ground truth laser scan depth using the Make3D Range Image dataset [122, 123]. We uniformly divide each image into 55×305 patches as [123]. The feature inputs for the first-layer depth estimation $\Psi_2 \in \mathbb{R}^{104}$ are features which capture texture, color and gradient properties of the patch. This is obtained by convolv-

ing the image with Laws' masks and computing the energy and Kurtosis over the patch along with the shape features as described by Saxena et al. [123].

We use a linear regression for the first-level and second-level instantiation of the depth estimation module. The output of the first-layer depth estimation $Z_2 \in \mathbb{R}_+$ is the predicted depth of each patch in the image. In order to feed the first-layer depth output to the second-layer classifiers, for the scene categorization and event categorization tasks, we use a vector with the predicted depth of all patches in the image; for the other tasks, we use the 1-dimensional predicted depth for the patch/pixel/bounding-box, etc.

Event Categorization: For event categorization, we classify an image into one of the 8 sports events as defined by Li et al. [92]: bocce, badminton, polo, rowing, snowboarding, croquet, sailing and rock-climbing. For evaluation, we compute the rate of incorrectly assigning an event label to an image. The feature inputs for the first-layer event classifier $\Psi_3 \in \mathbb{R}^{43}$ is a 43-dimensional feature vector, which includes the top 30 PCA projections of the 512-dimensional GIST features [136], the 12-dimension global color features (mean and variance of RGB and YCrCb color channels over the entire image), and a bias term.

We use a multi-class logistic classifier on each layer for event classification. The output of the first-layer event classifier $Z_3 \in \mathbb{R}^8$ is an 8-dimensional vector where each element represents the log-odd score of the corresponding image belonging to a event category. This 8-dimensional output is feed to each of the second-layer classifiers.

Saliency Detection. The goal of the saliency detection task is to classify each pixel in the image as either salient or non-salient. We use the saliency detection

dataset used by Achanta et. al. [2] for our experiments. The feature inputs for the first-layer saliency classifier $\Psi_4 \in \mathbb{R}^4$ includes the 3-dimensional color-offset features based on the Lab color space as described by Achanta et al. [2] and a bias term.

We use a logistic model for the saliency estimation classifiers on both layers. The output of the first-layer saliency classifier $Z_4 \in \mathbb{R}$ is the log-odd score of a pixel being salient. In order to feed the first-layer saliency detection output to the second-layer classifiers, for the scene categorization and event categorization tasks, we form a vector with the predicted saliency of all the pixels in the image; for the other tasks, we use the 1-dimensional average saliency for the corresponding pixel/patch/bounding-box.

Object Detection. We consider the following object categories: car, person, horse and cow. We use the train-set and test-set of PASCAL 2006 [27] for our experiments. Our object detection module builds on the part-based detector of Felzenszwalb et. al. [34]. We first generate 5 to 100 candidate windows for each image by applying the part-based detector with a low threshold (over-detection). The feature inputs for the first-layer object detection classifier $\Psi_5 \in \mathbb{R}^K$ are the HOG features extracted based on the candidate window as [18] plus the detection score from the part-based detector [34]. K depends on the number of scales to be considered and the size of the object template.

We learn an RBF-kernel SVM model as the first layer classifier. The classifier assigns each window a +1 or 0 label indicating whether the window belongs to the object or not. For the second-layer classifier, we learn a logistic model over the feature vector constituted by the outputs of all first-level tasks and the original HOG feature. We use average precision to quantitatively measure the

performance. The output of the first-layer object detection classifier $Z_5 \in \mathbb{R}^4$ are the estimated 0 or 1 labels for a region to belong to the 4 object categories we consider. In order to feed the first-layer object detection output to the second-layer classifiers, we first generate a detection map for each object. Pixels inside the estimated positive boxes are labeled as "+1", otherwise they are labeled as "0". For scene categorization and event categorization on the second layer, we feed all the elements on the map; for the other tasks, we use the 1-dimensional average value on the map for the corresponding pixel/patch/bounding-box.

Geometric labeling. The geometric labeling task refers to assigning each pixel to one of three geometric classes: support, vertical and sky, as defined by Hoiem et al. [57]. For evaluation, we compute the accuracy of assigning the correct geometric label to a pixel. The feature inputs for the first-layer geometry labeling classifier $\Psi_6 \in \mathbb{R}^{52}$ are the region-based features as described by Hoiem et al. [57].

We use the dataset and the algorithm by [57] as the first-layer geometric labeling module. To reduce the computation time, we avoid the multiple segmentations and instead use a single segmentation with 100 segments per image. We use a logistic model as the second-layer classifier. The output of the first-layer geometry classifier $Z_6 \in \mathbb{R}^3$ is a 3-dimensional vector with each element representing the log-odd score of the corresponding pixel belonging to a geometric category. In order to feed the first-layer geometry output to the second-layer classifiers, for scene/event categorization we form a vector with the predicted scores of all pixels; for the other tasks we use the 3-dimensional vector with each element representing the average scores for the corresponding pixel/patch/bounding-box.

Table 2.1: Summary of results for the SIX vision tasks. Our method improves performance in every single task. (Note: Bold face corresponds to our model performing equally with or better than state-of-the-art.)

Model	Event Categorization (% Accuracy)	Depth Estimation (RMSE in m)	Scene Categorization (% Accuracy)	Saliency Detection (% Accuracy)	Geometric Labeling (% Accuracy)	Object detection (% Average precision)				
						Car	Person	Horse	Cow	Mean
Images in testset	1579	400	2688	1000	300	2686				
Chance	22.5	24.6	22.5	50	33.3	-	-	-	-	-
Our base-model	71.8 (±0.8)	16.7 (±0.4)	83.8 (±0.2)	85.2 (±0.2)	86.2 (±0.2)	62.4	36.3	39.0	39.9	44.4
All-features-direct	72.7 (±0.8)	16.4 (±0.4)	83.8 (±0.4)	85.7 (±0.2)	87.0 (±0.6)	62.3	36.8	38.8	40.0	44.5
State-of-the-art	73.4	16.7 (MRF) [4]	83.8	82.5 (±0.2)	88.1	61.5	36.3	39.2	40.7	44.4
model (reported)	Li [92]	Saxena [123]	Torralba [135]	Achanta [2]	Hoiem [57]	Felzenswalb et. al. [33] (base)				
CCM [53]	73.3 (±1.6)	16.4 (±0.4)	83.8 (±0.6)	85.6 (±0.2)	87.0 (±0.6)	62.2	37.0	38.8	40.1	44.5
FE-CCM (unified)	**74.3 (±0.6)**	**15.5 (±0.2)**	**85.9 (±0.3)**	**86.2 (±0.2)**	**88.6 (±0.2)**	**63.2**	**37.6**	**40.1**	40.5	**45.4**
FE-CCM (one goal)	**74.2 (±0.8)**	**15.3 (±0.4)**	**85.8 (±0.5)**	**87.1 (±0.2)**	**88.6 (±0.3)**	**63.2**	**37.9**	**40.1**	**40.7**	**45.5**
FE-CCM (target specific)	**74.7 (±0.6)**	**15.2 (±0.2)**	**86.1 (±0.2)**	**87.6 (±0.2)**	**88.9 (±0.2)**	**63.2**	**38.0**	**40.1**	**40.7**	**45.5**

2.5 Experiments and Results

2.5.1 Experimental Setting

The proposed FE-CCM model is a unified model which jointly optimizes for all the sub-tasks. We believe this is a powerful algorithm in that, while independent efforts towards each sub-task have led to state-of-the-art algorithms that require intricate modeling for that specific sub-task, the proposed approach is a unified model which can beat the state-of-the-art performance in each sub-task and, can be seamlessly applied across different applications.

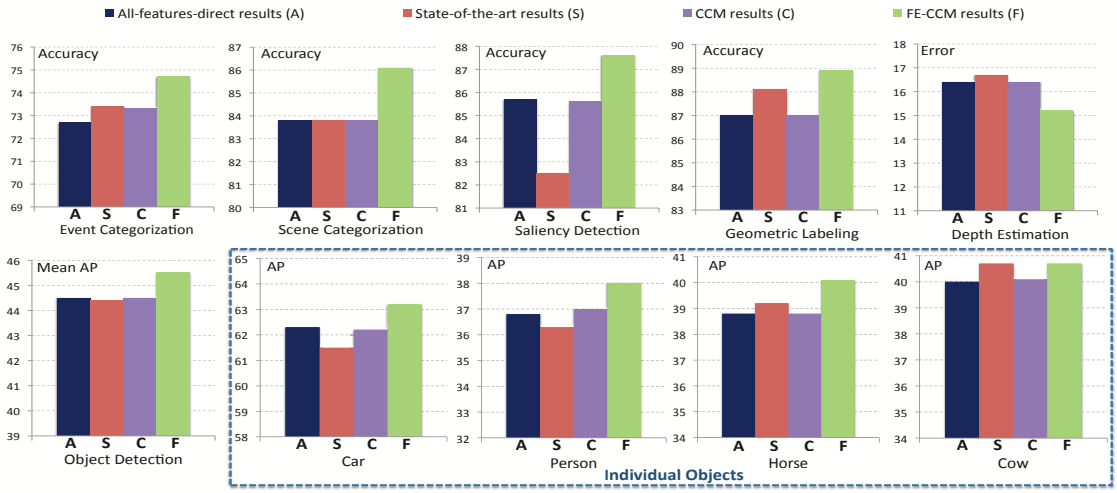


Figure 2.3: Results for the six tasks in scene understanding. Top: the performance for event categorization, scene categorization, saliency detection, geometric labeling, and depth estimation. Bottom: the average performance for object detection and the performance for the detection of individual object categories: car, person, horse, and cow. Each figure compares four methods: all-features-direct method, state-of-the-art methods, CCM, and the proposed FE-CCM method.

We evaluate our proposed method on combining six tasks introduced in Section 2.4. In our experiment, the training of FE-CCM takes 4-5 iterations. For each of the sub-tasks in each of the domains, we evaluate our performance on the standard dataset for that sub-task (and compare against the specifically designed state-of-the-art algorithm for that dataset). Note that, with such disjoint yet practical datasets, no image would have ground truth available for more than one task. Our model handles this well.

In experiment we evaluate the following algorithms as shown in Table 2.1,

- Base model: Our implementation (Section 2.4) of each sub-task, which serves as a base model for our FE-CCM. (The base model uses less information than state-of-the-art algorithms for some sub-tasks.)
- All-features-direct: A classifier that takes all the features of all sub-tasks, appends them together, and builds a separate classifier for each sub-task.
- State-of-the-art model: The state-of-the-art algorithm for each sub-task re-

spectively on that specific dataset.

- CCM: The cascaded classifier model by Heitz et al. [53], which we re-implement for six sub-tasks.
- FE-CCM (unified): This is our proposed model. Note that this is *one single model* which maximizes the joint likelihood of all the sub-tasks.
- FE-CCM (one goal): In this case, we have only one sub-task instantiated on the second layer, and the goal is to optimize the outputs of that sub-task. We train a specific one-goal FE-CCM for each sub-task.
- FE-CCM (target specific): In this case, we train a specific FE-CCM for each sub-task, by using cross-validation to estimate π_j's in Equation 2.23. Different values for π_j's result in different parameters learned for each FE-CCM.

Note that both CCM and All-features-direct use information from all sub-tasks, and state-of-the-art models also use carefully designed models that implicitly capture information from the other sub-tasks.

2.5.2 Datasets

The datasets used are mentioned in Section 2.4, and the number of test images in each dataset is shown in Table 2.1. For each dataset we use the same number of training images as the state-of-the-art algorithm (for comparison). We perform 6-fold cross validation on the whole model with 5 of 6 sub-tasks to evaluate the performance on each task. We do not do cross-validation on object detection as it is standard on the PASCAL 2006 [27] dataset (1277 train and 2686 test images respectively).

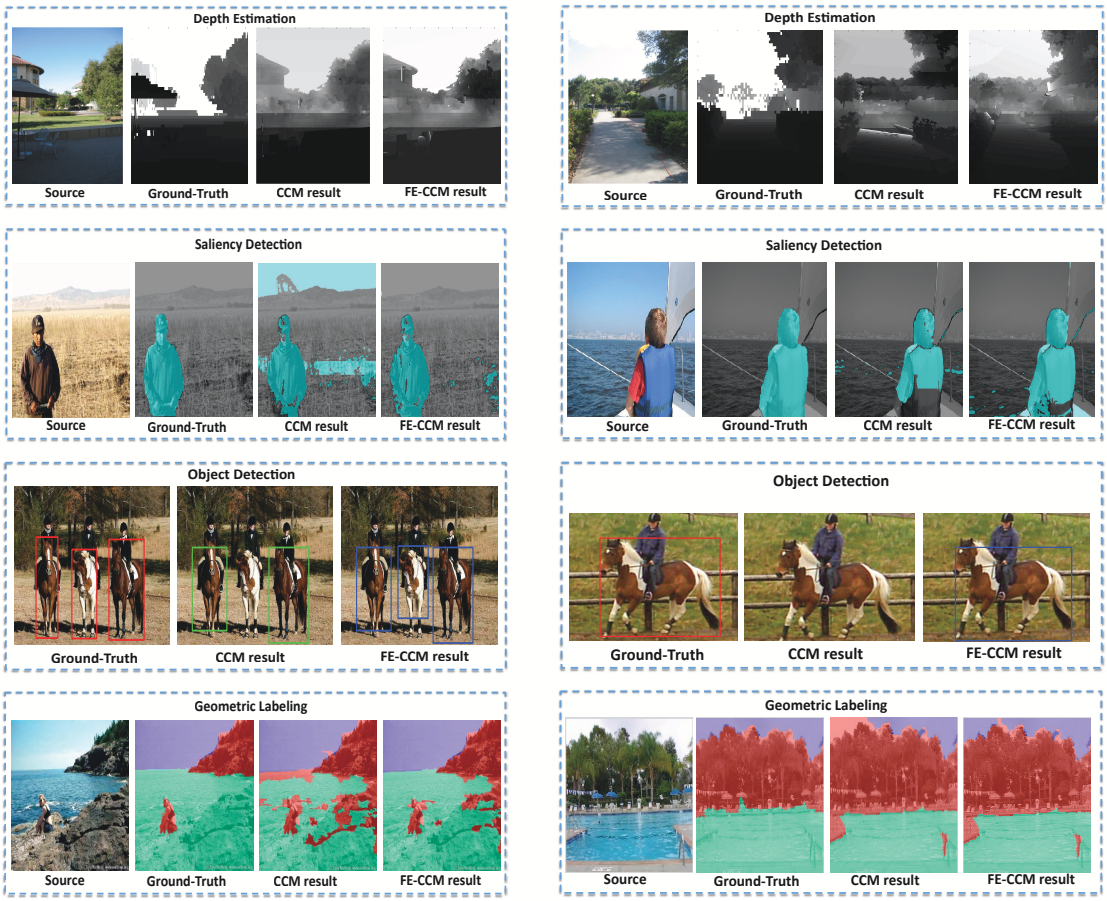


Figure 2.4: Results showing improvement using the proposed model. From top to bottom: Depth estimation, Saliency detection, Object detection, Geometric labeling. All depth maps in depth estimation are at the same scale (black means near and white means far); Salient region in saliency detection are indicated in cyan; Geometric labeling: Green=Support, Blue=Sky and Red=Vertical (Best viewed in color).

2.5.3 Results

To quantitatively evaluate our method for each of the sub-tasks, we consider the metrics appropriate to each of the six tasks in Section 2.4. Table 2.1 and Figure 2.3 show that FE-CCM not only beats state of the art in *all* the tasks but also does it jointly as *one single unified model*.

In detail, we see that all-features-direct improves over the base model because it uses features from all the tasks. The state-of-the-art classifiers improve

on the base model by explicitly hand-designing the task specific probabilistic model [92, 123] or by using adhoc methods to implicitly use information from other tasks [57]. Our FE-CCM model, which is a single model that was not given any manually designed task-specific insight, achieves greater improvement over the base model.

We also compare the three instantiations of FE-CCM in Table 2.1 (the last three rows). We observe that the target-specific FE-CCM achieves the best performance, by selecting a set of π_j's to optimize for each task independently. Though the unified FE-CCM achieves slightly worse performance, it jointly optmizes for all the tasks by training only one set of parameters. The performance of one-goal FE-CCM is less stable compared to the other two instantiations. It is mainly because the first-layer classifiers only gain feedback from the specific task on the second layer in one-goal FE-CCM, which easily causes overfitting.

We note that our target-specific FE-CCM, which is optimized for each task independently and achieves the best performance, is a more fair comparison to the state-of-the-art because each state-of-the-art model is trained specifically for the respective task. Furthermore, Figure 2.3 shows the results for CCM (which is a cascade without feedback information) and all-features-direct (which uses features from all the tasks). This indicates that the improvement is strictly due to the proposed feedback and not just because of having more information.

We show some visual improvements due to the proposed FE-CCM in Figure 2.4. In comparison to CCM, FE-CCM leads to better depth estimation of the sky and the ground, and it leads to better coverage and accurate labeling of the salient region in the image, and it also leads to better geometric labeling and

[4]The state-of-the-art method for depth estimation in [123] follows a slightly different testing procedure. In that case, our target-specific FE-CCM method achieves $RMSE = 15.3$.

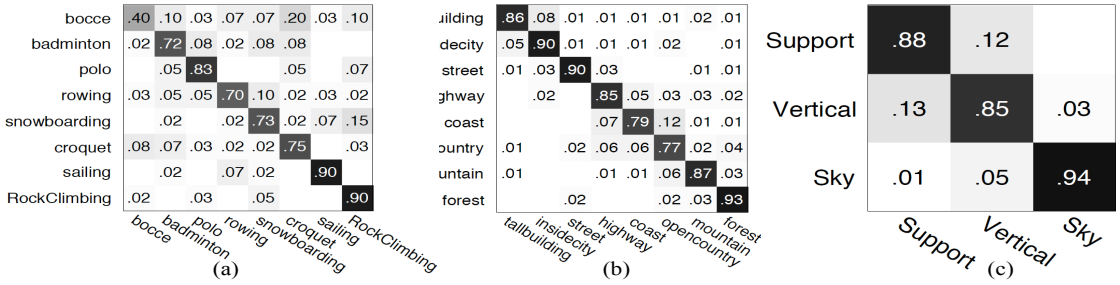


Figure 2.5: Confusion matrix for (a) Event categorization; (b) Scene categorization; (c) Geometric labeling. All the results are gained with the proposed FE-CCM method. The average accuracy achieved by the proposed FE-CCM model outperforms the state-of-the-art methods for each of these tasks, as listed in Table 2.1.

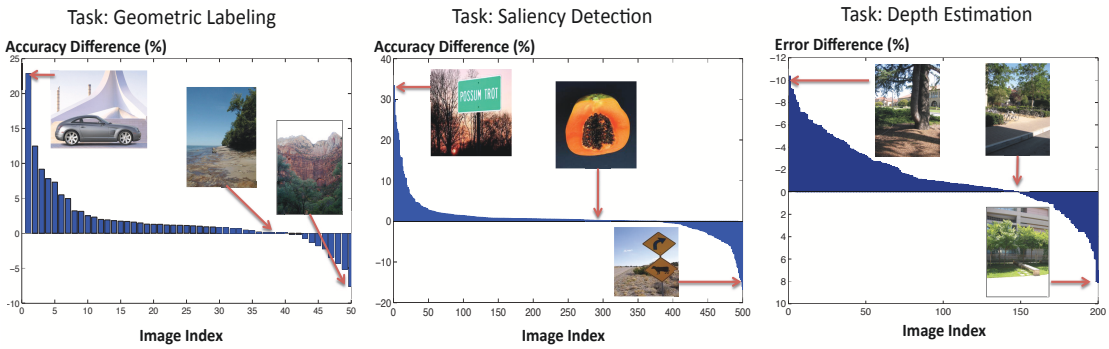


Figure 2.6: Performance difference between the proposed unified FE-CCM method and the all-features-direct method for each test image, respectively for the tasks of geometric labeling, saliency detection, depth estimation, on one of the cross-validation folds.

object detection. Figure 2.5 also provides the confusion matrices for the three tasks: scene categorization, event categorization, geometric labeling.

Figure 2.6 provides scatter plots of the performance difference for each image between the unified FE-CCM method and the all-features-direct method, respectively for the tasks of geometric labeling, saliency detection, and depth estimation. We note that for all three tasks, the unified FE-CCM outperforms the all-features-direct method on most images. For geometric labeling and saliency detection, the improvement from the unified FE-CCM method is mainly due to

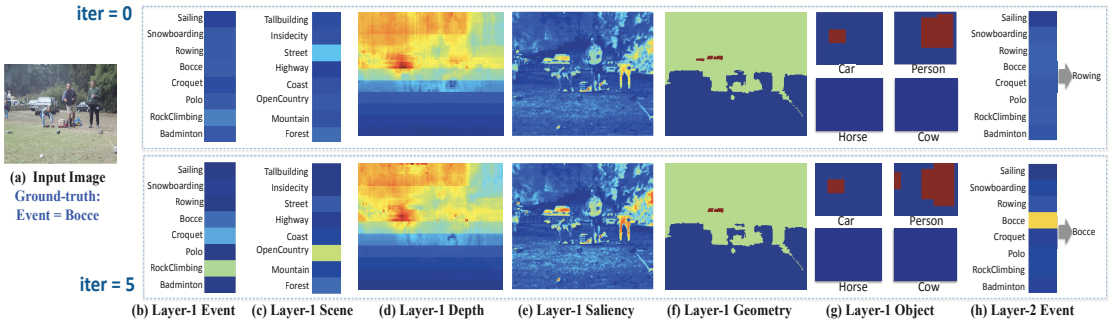


Figure 2.7: Illustration of the FE-CCM first-layer outputs for a single image. (a) the input image from the sports-event dataset. Its groundtruth event label is "Bocce". (b-g)Outputs of the first-layer classifiers, at initialization (top row) and at the 5^{th} iteration (bottom row). (h) Outputs of the second-layer event classifier. Note that at initialization the first-layer classifiers are trained using ground-truth labels, i.e. the same as CCM. In (b)(c)(d)(e)(h), Red=High-value, Blue=low-value. In (f), Blue=Ground, Green=Vertical, Red=Sky. In (g) Red=Object Presence. (Best viewed in color.)

Table 2.2: Summary of results for combining scene categorization and object detection, with partially-labeled datasets and fully-labeled datasets.

Model	Scene Categorization (% accuracy) partial-labeled / full-labeled	Object Detection (% mean AP) partial-labeled / full-labeled
Our base-model	45.6 / 47.5	67.6 / 70.7
All features direct	46.8 / 49.1	71.2 / 72.5
CCM [53]	50.8 / 52.3	74.0 / 76.1
FE-CCM (unified)	**54.2 / 54.8**	**77.5 / 77.9**

large improvements on some images. For depth estimation, the improvement is scattered over many images.

The cause of improvement. We have shown improvements of FE-CCM in Table 2.1 under the situation of heterogeneous datasets. The improvement can be caused by one or both of the following reasons: (1) the feedback process finds better error modes for the first-layer classifiers; (2) the feedback generates additional "labels" to retrain the first-layer classifiers. In order to analyze this,

we consider the two tasks of scene recognition and object detection on the DS1 dataset in [53], which contains ground-truth labels for both the tasks. We compare the various methods under two settings: (1) train with the fully-labeled data; (2) train with only the scene labels for one half of the training data and only the object labels for the second half. Table 2.2 compares the performance of training with partially-labeled datasets and the performance of different methods under these two settings. The experiments are performed using 5-fold cross validation. The unified FE-CCM method outperforms the other methods under both partially-labeled and fully-labeled situations. We note that all methods listed perform better when full labels are provided. In fact, FE-CCM achieves close performance in both settings. We also note that the FE-CCM method trained with partially-labeled datasets outperforms the CCM method trained with fully-labeled datasets, which indicates that the improvement achieved by the FE-CCM method is not simply from generating more labels for training the first-layer classifiers, but also due to finding useful modes for the first-layer classifiers.

Figure 2.7 illustrates the first-layer outputs of a test image, respectively at initialization and at the 5^{th} iteration. Our initialization is the same as CCM, i.e., using ground-truth labels to train the first-layer classifiers. We note that with feedback, the first-layer output shifts to focus on more meaningful modes, e.g., At initialization, the event classifier has widespread confusion with other categories. With feedback, the event classifier turns to be confused with only the 'rock-climbing' and 'croquet' events which are more similar to 'bocce'. Moreover, the first-layer scene, depth, and object classifiers also give more meaningful predictions while trained with feedback. With better first-layer predictions, our FE-CCM correctly classifies the event as 'bocce', while CCM misclassifies

it as 'rowing'. Furthermore, Figure 2.8 provides a specific example on how the performance of the first-layer horse detection classifier changes with feedback. We note that, it is not obvious whether the performance is better or not with feedback. Instead, the feedback changes the error mode of the classifier: it prefers the precision to be better when the recall is high, and sacrifices the precision when the recall is low.

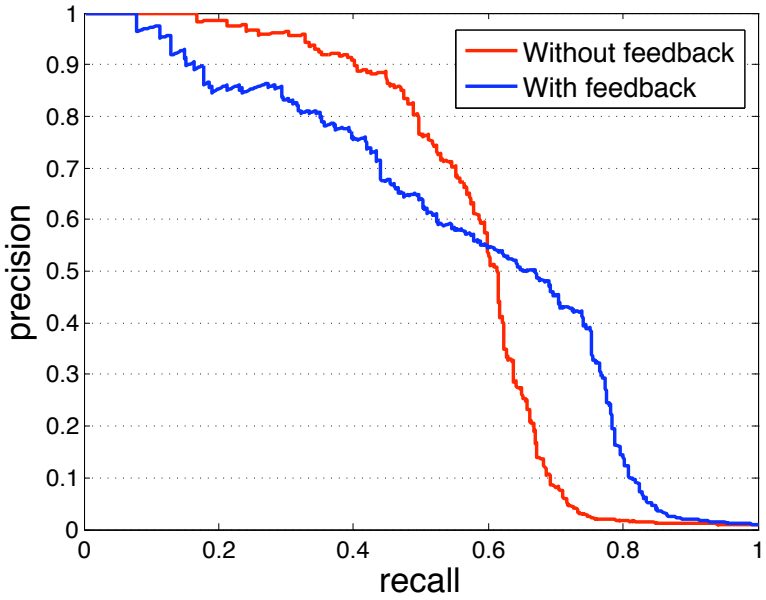


Figure 2.8: The difference of the detection performance for the first-layer horse detection classifier without feedback and with feedback.

2.5.4 Discussion

FE-CCM allows each classifier in the second layer to learn which information from the other first-layer sub-tasks is useful, and this can be seen in the learned weights Ω for the second-layer. We provide a visualization of the weights for the six vision tasks in Figure 2.9(a). We see that the model agrees with our intuitions that large weights are assigned to the outputs of the same task from the first layer classifier (see the large weights assigned to the diagonals in the

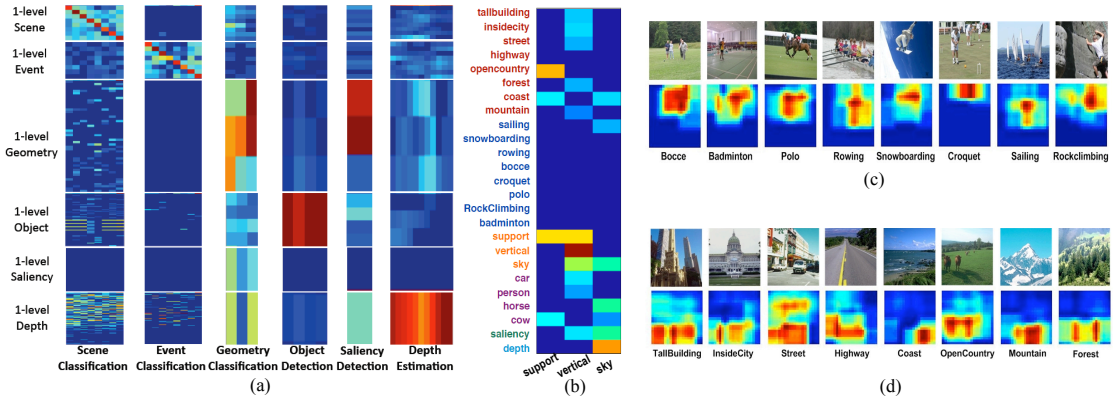


Figure 2.9: (a) The *absolute* values of the weight vectors for second-level classifiers, i.e. ω. Each column shows the contribution of the various tasks towards a certain task. (b) Detailed illustration of the *positive* values in the weight vector for a second-level geometric classifier. (c)(d) Illustration of the importance of depths in different regions for predicting different events (c) and scenes (d). An example image for each class is also shown above the map of the weights. (Note: Blue is low and Red is high. Best viewed in Color).

categorization tasks), though saliency detection is an exception which depends more on its original features (not shown here) and the geometric labeling output. We also observe that the weights are sparse. This is an advantage of our approach since the algorithm automatically figures out which outputs from the first level classifiers are useful for the second level classifier to achieve the best performance.

Figure 2.9(b) provides a closer look to the positive weights given to the various outputs for a second-level geometric classifier. We observe that large positive weights are assigned to "mountain", "forest", "tall building", etc. for supporting the geometric class "vertical", and similarly "coast", "sailing" and "depth" for supporting the "sky" class. These illustrate some of the relationships the model learns automatically without any manual intricate modeling.

Figure 2.9(c) visualizes the weights given to the depth attributes (first-layer depth outputs) for the task of event categorization. Figure 2.9(d) shows the same

for the task of scene categorization. We see that the depth plays an important role in these tasks. In Figure 2.9(c), we observe that most event categories rely on the middle part of the image, where the main objects of the event are often located. E.g., most of the "polo" images have horses and people in the middle of the image while many "snowboarding" images have people jumping in the upper-middle part. For scene categorization, most of the scene categories (e.g., coast, mountain, open country) have sky in the top part, which is not as discriminative as the bottom part. In scene categories of tall buildings and street, the upper part of the street consists of buildings, which discriminates these two categories from the others. Not surprisingly, our method had automatically figured this out (see Figure 2.9(d)).

Stability of the FE-CCM algorithm: In the earlier section, we have presented results for six sub-tasks. In order to find out how our method scales with different combination and number of sub-tasks, we have tried several combinations, and in each case we get consistent improvement in each sub-task. For example, in our preliminary experiments, combining scene categorization and object detection gives us 15.4% and 10.2% respective improvements (Table 2.2). We then combined four tasks: event categorization, scene categorization, depth estimation, and saliency detection, and got improvements in all these sub-tasks [65].

Initialization of the FE-CCM algorithm: Our algorithm uses the ground-truth of the specific tasks to initialize the hidden variables on the first layer. In experiment, it seem to be a good initialization, as the semantic concepts in the labels can be considered as strong priors for the learning of the first-layer classifiers. We also compare our initialization with the random initialization method, con-

Table 2.3: The performance of different methods: Base, CCM, FE-CCM with ground-truth initialization, and FE-CCM with random initialization.

	Scene Categorization (% accuracy)	Depth Estimation (RMSE in m)
Base	83.8	16.7
CCM	83.8	16.5
FE-CCM: random init	84.8	16.0
FE-CCM: groundtruth init	**85.8**	**15.6**

sidering a simpler framework with two tasks: scene categorization and depth estimation. The results are shown in Table 2.3. The results indicate that our initialization is more effective than random initialization.

2.6 Summary

We propose a method for combining existing classifiers for different but related tasks in scene understanding. We only consider the individual classifiers as a 'black-box' (thus not needing to know the inner workings of the classifier) and propose learning techniques for combining them (thus not needing to know how to combine the tasks). Our method introduces feedback in the training process from the later stage to the earlier one, so that a later classifier can provide the earlier classifiers information about what error modes to focus on, or what can be ignored without hurting the joint performance.

Our extensive experiments show that our unified model (a single FE-CCM trained for all the sub-tasks) improves performance across *all* the sub-tasks considered over the respective state-of-the-art classifiers. We show that this was the result of our feedback process. The classifier actually learns meaningful re-

lationships between the tasks automatically.

This work has first appeared in the following publications: Li, Kowdle, Saxena, Chen [65, 83, 84].

CHAPTER 3

SHARING CONTEXTUAL PARAMETERS BETWEEN TASKS

As we discuss earlier, holistic scene understanding involves many sub-tasks, such as scene categorization, depth estimation, object detection and so on. Some of the tasks are even location-dependent, which encourages us to train different classifiers for the same property on different regions of the image. Therefore, if we scale up the number of tasks in the two-layer cascaded classification model, the number of parameters to be trained would become quite large, and training the large amount of classifiers with limited training data would not be effective. Therefore, in this chapter, we look into another direction: how can we train such a large amount of parameters robustly in the CCM structure? To do this, we consider modeling the interactions between the parameters of the target tasks. We propose modeling such interactions by an undirected graph called θ-MRF, where the nodes represent the parameters for each task and the edges represent the interaction between the parameters.

3.1 Introduction

Most scene understanding tasks (e.g., object detection, depth estimation, etc.) require that we exploit contextual information in addition to the local features for predicting the labels. For example, a region is more likely to be labeled as a car if the region below is labeled as road. I.e., we have to consider information in a larger area around the region of interest. Furthermore, the location of the region in the image could also have a large effect on its label, and on how it

depends on the neighboring regions. For example, one would look for sky or clouds when looking for an airplane; however if one sees grass or a runway, then there may still be an airplane (e.g., when the airplane is on the ground)—here the contextual dependence of the airplane classifier changes based on object's location in the image.

We can capture such contextual information by using features from *all* the regions in the image, and then also train a specific classifier of each spatial location for each object category. However, the dimensionality of the feature space would become quite large,[1] and training a classifier with limited training data would not be effective. In such a case, one could reduce the amount of context captured to prevent overfitting. For example, some recent works [53, 91, 101] use context by encoding input features, but are limited by the amount of context area they can handle.

In this part of work, we do not want to eliminate the amount of context captured. We therefore keep the large number of parameters, and model the interaction between the parameters of the classifiers at different locations and different tasks. For example, the parameters of two neighboring locations are similar. The key contribution of this part is to note that two parameters may not ascribe a directionality to the interaction between them. These interactions are sparse, and we represent these interactions as an undirected graph where the nodes represent the parameters for each location (for each task) and the edges represent the interaction between the parameters. We call this representation a *θ-MRF*, i.e., a *Markov Random Field over the parameters*. This idea is, in principle,

[1] As an example, consider the problem of object detection with many categories: we have 107 object categories which may occur in any spatial location in the image. Even if we group the regions into 64 (8×8) spatial locations, the total number of parameters will be $107 * 64 * K$ (for K features each). This is rather large, e.g., in our multi-class object detection task this number would be about 47.6 million (see Section 3.4).

complementary to previous works that capture context by capturing the correlation between the labels. Note that our goal is not to directly compare against such models. Instead, we want to answer the question: *How far can we go with just modeling the interactions between the parameters?*

The edges in our θ-MRF not only connect spatial neighbors but also semantic neighbors. In particular, if two tasks are highly correlated, their parameters given to the same image context should be similar. For example, oven is often next to the dishwasher (in a kitchen scene), therefore they should share similar context, indicating that they can share their parameters. These semantic interactions between the parameters from different tasks also follow the undirected graph. Just like object labels are often modeled as conditionally independent of other non-contextual objects given the important context, the corresponding parameters can also be modeled similarly.

There has been a large body of work that capture contextual information in many different ways which are often complementary to ours. These methods range from capturing the correlation between labels using a graphical model to introduce different types of priors on the labels (based on location, prior knowledge, etc.). For example, a graphical model (directed or undirected) is often used to model the dependency between different labels [42, 47, 68, 113]. Informative priors on the labels are also commonly used to improve performance (e.g., [137]). Some previous works enforce priors on the parameters as a directed graph [90, 128], but our model offers a different and perhaps a more relevant perspective than a directed model, in terms of the independence properties modeled.

We extensively evaluate our method on two different settings. First, we con-

sider the task of labeling 107 object categories in the SUN09 dataset, and show that our method gets better performance than the state-of-the-art methods even when with simple regression as the learning model. Second, we consider the multiple tasks of scene categorization, depth estimation and geometry labeling, and again show that our method gets comparable or better performance than the state-of-the-art methods when we use our method with simple regression. Furthermore, we show that our performance is much higher as compared to just using other methods of putting priors on the parameters.

3.2 Related Work

There is a large body of work that leverages contextual information. We possibly cannot do justice to literature, but we mention a few here. Various sources of context have been explored, ranging from the global scene layout, interactions between regions to local features. To incorporate scene-level information, Torralba et al. [137] use the statistics of low-level features across the entire scene to prime object detection. Hoiem et al. [57] and Saxena et al. [123] use 3D scene information to provide priors on potential object locations. Li et al. [90] propose a hierarchical model to make use of contextual information between tasks on different levels. There are also generic approaches [53, 83] that leverage related tasks to boost the overall performance, without requiring considerate insight into specific tasks.

Many works also model context to capture the local interactions between neighboring regions [54, 67, 94], objects [35, 148], or both [7, 24, 41]. Object co-occurence statistics have also been captured in several ways, e.g., using a

CRF [42, 47, 113]. Desai et al. [22] combine individual classifiers by considering spatial interactions between the object detections, and solve a unified multi-class object detection problem through a structured discriminative approach. Other ways to share information across categories include sharing representations [30, 70], sharing training examples between categories [36, 98], sharing parameters [59, 62], and so on. Our work in this chapter lies in the category of sharing parameters, aiming at capturing the dependencies in the parameters for relevant vision applications.

There are several regularization methods when the number of parameters is quite large, e.g., based on L2 norms [11] and Lasso shrinkage methods [117]. Liang et al. [93] present an asymptotic analysis of smooth regularizers. Recent works [5, 44, 58, 59] place interesting priors on parameters. Jalali et al. [59] do multi-task learning by expressing the parameters as a sum of two parts: shared and specific to the task, which combines the l_∞ penalty and l_1 penalty to get block-sparse and element-wise sparse components in the parameters. Negahban and Wainright [104] provide analysis of when $l_{1,\infty}$ norm could be useful. Kim and Xing [62] use a tree to construct the hierarchy of multi-task outputs, and then use the tree-guided group lasso to regularize the multi-task regression. In contemporary work [120], Salakhutdinov et al. learn a hierarchy to share the hierarchical parameters for the object appearance models. Our work in this chapter is motivated by this direction of work, and our focus is to capture spatial and semantic sharing in parameters using undirected graphical models that have appropriate independence properties.

Bayesian priors over parameters are also quite commonly used. For example, [8] uses Dirichlet priors for parameters of a multinomial and normal

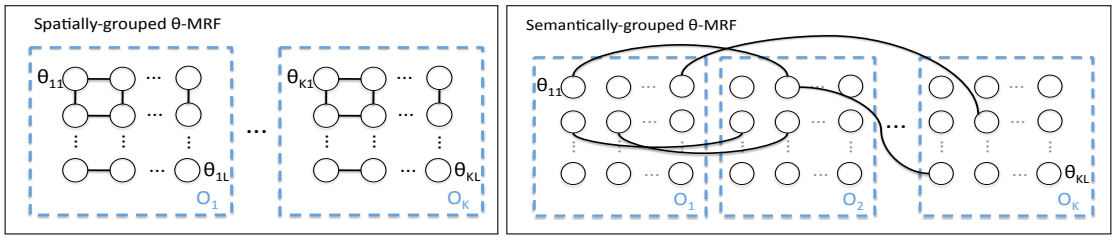


Figure 3.1: The proposed θ-MRF graph with spatial and semantic interaction structure.

distribution respectively. In fact, there is a huge body of work on using non-informative priors distributions over parameters [9]—this is particularly useful when the amount of data is not enough to train the parameters. If all the distributions involved (including the prior distribution) are Gaussian, the parameters follow certain useful statistical hyper Markov properties [20, 52, 116]. In applications, [128] considers capturing relationships between the object categories using a Dirichlet prior on the parameters. [49] considers putting posterior sparsity on the parameters instead of parameter sparsity. [25] present a method to learn hyperparameters for CRF-type models. Most of these methods express the prior as another distribution with hyper-parameters—one can view this as a directed graphical model over the parameters. On the other hand, we express relationships between two parameters of the distribution, which does not necessarily involve hyper parameters. This also allows us to capture interesting independence properties.

3.3 Approach: θ-MRF

In order to give better intuition, we use the multi-class object detection task as an illustrative example. (Later we will describe and apply it to other scene understanding problems.) Let us consider the K-class object detection. We uni-

formly divide an image into L grids. We then have a binary classifier, whose output is $y_{k,\ell}^{(n)} \in \{0,1\}$ that indicates the presence of the k^{th} object at the ℓ^{th} grid in the n^{th} image. Let $x^{(n)}$ be the features (or attributes) extracted from n^{th} image, and let the parameters of the classifier be $\theta_{k,\ell}$. Let $\Theta_k = (\theta_{k,1}, \cdots, \theta_{k,L})$ and let $\mathbf{\Theta}$ be the set $\{\Theta_k\}$, $k = 1, \ldots, K$.

Let $P(y_{k,\ell}|x^{(n)}, \theta_{k,\ell})$ be the probability of the output given the input features and the parameters. In order to find the classifier parameters, one typically solves an optimization problem, such as:

$$\underset{\mathbf{\Theta}}{\text{minimize}} \qquad \sum_{n} \sum_{k,l} -\log P(y_{k,\ell}|x^{(n)}, \theta_{k,\ell}) + R(\mathbf{\Theta}) \tag{3.1}$$

where $R(\mathbf{\Theta})$ is a regularization term (e.g., $\lambda\|\mathbf{\Theta}\|_2^2$ with λ as a tuning parameter) (In Bayesian view, it is a prior on the parameters that could be informative or non-informative.) Let us use $J(\theta_{k,\ell}) = -\log P(y_{k,\ell}|x^{(n)}, \theta_{k,\ell})$ to indicate the cost of the data dependent term $\theta_{k,\ell}$. The exact form of $J(\theta_{k,\ell})$ would depend on the particular learning model being used over the labels y's. For example, for logistic regression it would be $J(\theta_{k,\ell}) = -\log\Big((\frac{1}{1+e^{-\theta_{k,\ell}^T x^{(n)}}})^{y_{k,\ell}}(1 - \frac{1}{1+e^{-\theta_{k,\ell}^T x^{(n)}}})^{(1-y_{k,\ell})}\Big)$. Motivated by the earlier discussion, we want to model the interactions between the parameters of the different classification models, indexed by $\{k, \ell\}$ that we merge into one index $\{m\}$.

In this part of work, we represent these interactions as an undirected graph $\mathcal{G}$ where each node m represents the parameters θ_m. The edges $\mathcal{E}$ in the this graph would represent the interaction between two sets of parameters θ_i and θ_j. These interactions are often sparse. We call this graph θ-MRF. Eq. 3.1 can now be viewed as optimizing the energy function of the MRF *over the parameters*. I.e.,

$$\underset{\mathbf{\Theta}}{\text{minimize}} \sum_{m \in \mathcal{G}} J(\theta_m) + \sum_{i,j \in \mathcal{E}} R(\theta_i, \theta_j) \tag{3.2}$$

where $J(\theta_m)$ is now the node potential, and the term $R(\theta_i, \theta_j)$ corresponds to the edge potentials. Note this idea of MRF is quite complementary of other modeling structures one may impose over y's—which may itself be an MRF. This θ-MRF is different from the label-based MRFs whose variables y's are often in low-dimension. In our parameter-based MRF, each node constitutes high-dimensional variables θ_m. One nice property of having an MRF over parameters is that there is no increase in complexity of the inference problem.

In previous work (also see Section 3.2), several priors have been used on the parameters. Such priors are often in the form of imposing a distribution with some other hyper parameters—this corresponds to a directed model on the $\mathbf{\Theta}$ and in some application scenarios they may not be able to express the desired conditional independence properties and therefore may be sub-optimal. Our θ-MRF is largely a non-informative prior, and also corresponds to some regularization methods. See Section 3.5 for experimental comparisons with different forms of priors. Having presented this general notion of θ-MRF, we will now describe two types of interactions that it models well in the following.

3.3.1 Spatial Interactions

Intuitively the parameters of the classifiers at neighboring spatial regions (for the same object category) should share their parameters. To model this type of interactions between parameters, we introduce edges on the θ-MRF that connect the spatially neighboring nodes, as shown in Figure 3.1-left. Note that the spatial edges only couple the parameters of the same task together. This type of

edge does not exist across tasks. We define the edge potential as follows.

$$R(\theta_i, \theta_j) = \begin{cases} \lambda_{\text{spt}} \|\theta_i - \theta_j\|_p & \text{if } \theta_i \text{ and } \theta_j \text{ are spatial neighbors for a task} \\ 0 & \text{otherwise} \end{cases}$$

where λ_{spt} is a tuning factor for the spatial interactions. When $p \geq 1$, this potential has the nice property of being convex. Note that such a potential has been extensively used in an MRF over labels, e.g., [123]. Note that this potential does not make the original learning problem in Equation 3.1 any "harder." In fact, if the original objective $J(\theta)$ is convex, then the overall problem still remains convex. In this chapter, we consider $p = 1$ and $p = 2$.

In addition to connecting the parameters for neighboring locations, we also encourage the sharing between the elements of a parameter vector that correspond to spatially neighboring *inputs*. The intuition is described in the following example. Assume we have the presence of the object "road" at the different regions of an image as attributes. In order to learn a car detector with these attributes as inputs, we would like to give similar high-weights to the neighboring regions in the car detector output. We call this **source-based spatial grouping**, as compared to **target-based spatial grouping** that we described in the previous paragraph. We found that this also gives us a contextual map (i.e., parameters that map the feature/attributes in the neighboring regions) that is more spatially structured. This interaction happens within the same node in the graph, therefore it is equivalent to adding an extra term to the node potential on the θ-MRF.

$$J_{\text{new}}(\theta_m) = J(\theta_m) + \lambda_{\text{src}} \sum_{t_1} \sum_{t_2 \in Nr(t_1)} \|\theta_m^{t_1} - \theta_m^{t_2}\|_p \tag{3.3}$$

where $\theta_m^{t_1}$ and $\theta_m^{t_2}$ corresponds the weights given to the t_1^{th} and the t_2^{th} feature inputs. $t_2 \in Nr(t_1)$ means that the respective features are the same type of attributes form neighboring regions. Equation 3.3 can be reformed as $J_{\text{new}}(\theta_m) =$

$J(\theta_m) + \lambda_{\text{src}}\|\mathcal{T}\theta_m\|_p$, where $\mathcal{T}$ indicates the linear transform matrix that computes the difference in the neighbors. λ_{src} is a tuning factor for the source interactions.

3.3.2 Semantic Interactions

We not only connect the parameters for spatial neighbors of the same task, but also consider the semantic neighbors across tasks. Motivated by the conditional independency in the object labels which suggests that given the important context the presence of an object is independent of other non-contextual objects, we can encode such properties in our θ-MRF. For example, the road often appears below the car. Note that in our framework we have the road classifier and the car classifier take the same features as input, which are extracted from all regions of the images to capture long-range context. Since the high concurrence of these two objects, their corresponding detectors should be activated simultaneously. Therefore, the parameter for detecting "road" at a bottom region of the image, can partly share with the parameter for detecting "car" above the bottom region. Assume we already know the dependency between the objects, we introduce the semantic edge potential of the θ-MRF, as shown in Figure 3.1-right.

$$R(\theta_i, \theta_j) = \begin{cases} \lambda_{\text{smn}} w_{ij} \|\theta_i - \theta_j\|_p & \text{if } \theta_i \text{ and } \theta_j \text{ are semantic neighbors} \\ 0 & \text{otherwise} \end{cases}$$

where w_{ij} indicates the strength of the semantic dependency between these two parameters and λ_{smn} is a tuning factor for the semantic interactions. In the following we discuss how to find the semantic connections and the weights w's.

Finding the semantic neighbors. We first calculate the positive correlations between the tasks from the ground-truth training data. If two tasks are highly positively correlated, they are likely to share some of the parameters. In order to model how they share parameters, we model the relative spatial relationship between the positive outputs of the two tasks. For example, assume we have two highly co-occuring object categories, indexed by k_1 and k_2. From the training data, we learn the relative spatial distribution map of the presence of the k_2^{th} object, given the k_1^{th} object in the center. We then find out the top M highest response regions on the map, each of which has a relative location $\Delta\ell$ and co-occuring response w. Therefore, the parameters of the k_2^{th} object that satisfy these relative locations, have semantic edges with θ_{k_1,l_1}.

3.3.3 Computational Efficiency

$R(\mathbf{\Theta})$ couples the different independent parameters. Typically, the total number of parameters is quite large in an application (e.g., 47.6 million in one of our applications, see Section 3.4). Running an optimization algorithm jointly on all the parameters would either not be feasible or have very slow convergence in practice. Since the parameters follow conditional independence assumptions and also follow a nice topological structure, we can optimize more connected subsets of the parameters separately, and then iterate. These separate sub-problems can also run in parallel. In our implementation, $R(\mathbf{\Theta})$'s and $J(\theta_m)$ are convex, and such a decomposed algorithm for optimizing the parameters is guaranteed to converge to the global optima [10]. In the two experiment settings we consider, the training of our algorithm takes 6-7 hours for object detection and 3-4 hours for multi-task cascade. In the testing stage, the complexity of our inference is no

more than constant times of the complexity of inference of an individual classier. Furthermore, the inference for different classiers can be easily parallelized. For example, a base object detector [33] takes about 1.5 second to output results for an image. Our algorithm, taking the outputs of the base detectors as input, only requires an overhead of less than 0.2 second.

3.4 Implementation

We apply our θ-MRF on two different settings: 1) object detection on the SUN09 dataset [15]; 2) multiple scene understanding tasks (scene categorization, geometric labeling, depth estimation), comparing to the cascaded classification models (CCM) [53, 83].

Object Detection. The task of object detection is to recognize and localize objects of interest in an image. We use the SUN 09 dataset introduced in [15], which has 4,367 training images and 4,317 test images. Choi et al. [15] use an additional set of 26,000 images to training baseline detectors [33], and select 107 object categories to evaluate their contextual model. We follow the *same settings* as [15], i.e., we use the same baseline object detector outputs as the attribute inputs for our algorithm, the same training/testing data, and the same evaluation metrics. For evaluation, a predicted bounding box is considered correct if it overlaps the ground-truth bounding box (in the intersection/union sense) by more than 50%. We compute the average precision (AP) of the precision-recall curve for each category, and compute the mean AP across categories as the overall performance.

We use each of the baseline object detectors to produce a 8×8 detection map,

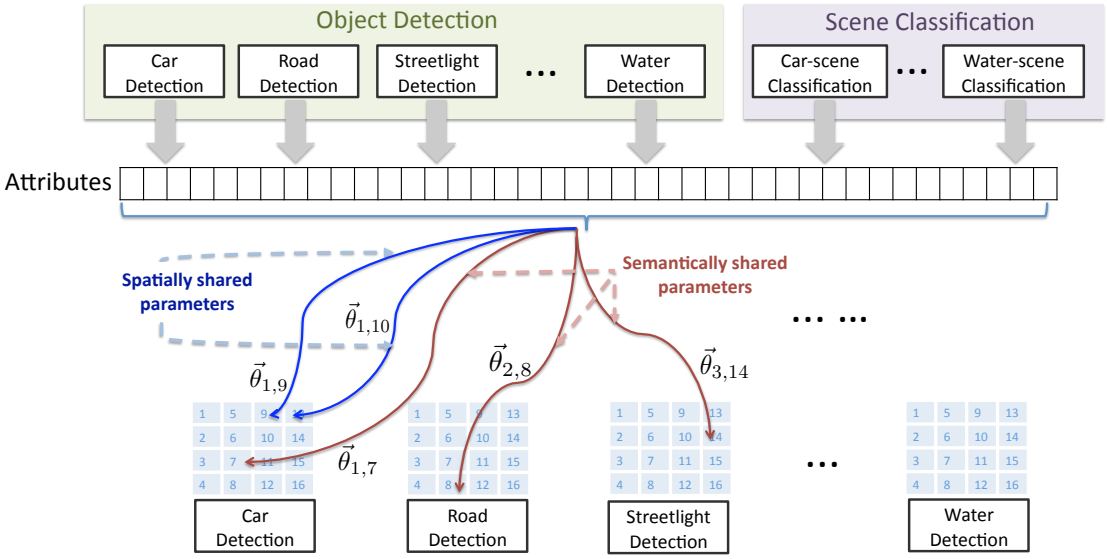


Figure 3.2: An instantiation of the proposed algorithm for the object recognition tasks on SUN09 dataset.

with each element indicating the confidence (between 0 and 1) of the object's presence at the respective region. We also define 107 scene categories, where the $i^{th}(i = 1, \ldots, 107)$ scene category indicates the type of scene containing the i^{th} object category. We train a logistic regression classifier for each scene category. The 107 8×8 object maps and the 107 scene classifier outputs together form a 6955-dimension feature vector, as the attribute inputs for our algorithm. The setup is shown in Figure 3.2.

We divide an image into 8×8 regions. Our algorithm learns a region-specific contextual model for each object category, resulting in a specific classifier of each region for each category. The 8×8 division is determined based on the criteria that more than 70% of the training data contain bounding boxes no smaller than a single grid. We use a linear model for each classifier. So we have $6955 * 8 * 8 * 107 = 47627840$ parameter dimensions in total. Our θ-MRF captures the independencies between these parameters based on location and semantics. For the l^{th} region, it is labeled as positive for the k^{th} object category if it satisfies:

$\text{overlap}(O_k, R_l)/\min(\text{area}(R_l), \text{area}(O_k)) > 0.3$, where O_k means a bounding-box instantiation of the k^{th} object category and R_l means the l^{th} grid cell. Negative examples are sampled from the false positives of the baseline detectors. We apply the trained classifiers to the test images, and gain the object detection maps. To create bounding-box based results, we use the candidate bounding boxes created by the baseline detectors, and average the scores gained from our algorithm within the bounding box as the confidence score for the candidate.

Multiple Scene Understanding Tasks. We consider the task of estimating different types of labels in a scene: scene categorization, geometry labeling, and depth estimation. We compose these three tasks in the feed-forward cascaded classification models (CCM) [53]. CCM creates repeated instantiations of each classifier on multiple layers of a cascade, where the latter-layer classifiers take the outputs of the previous-layer classifiers as input. The previous CCM algorithms [53, 83] consider sharing information across tasks, but do not consider the sharing between categories or between different spatial regions within a task. Here we introduce the semantically-grouped regularization to scene categorization, and the spatially-grouped regularization to depth and geometry estimation.

For the three tasks we consider, we use the same datasets and 2-layer settings as [83]. For scene categorization, we classify 8 different categories on the MIT outdoor scene dataset [105]. We consider two semantic groups: man-made (tall building, inside city, street, highway) and natural (coast, open-country, mountain and forest). Semantic edges are introduced between the parameters within each group. We train a logistic classifier for each scene category. This gives us a total of 8 parameter vectors for scene categorization task. We evaluate the per-

formance by measuring the accuracy of assigning the correct scene label to an image.

For depth estimation, we train a specific linear regression model for every region of the image (with uniformly divided 11×10 regions), and incorporate the spatial grouping on both the second-layer inputs and outputs. This gives us a total of 110 parameter vectors for the depth estimation task. We evaluate the performance by computing the root mean square error of the estimated depth with respect to ground truth laser scan depth using the Make3D Range Image dataset [123].

For geometry labeling, We use the dataset and the algorithm by [57] as the first-layer geometric labeling module, and use a single segmentation with about 100 segments/image. On the second-layer, we train a logistic regression classifier for every region of the image (with uniformly divided 16×16 regions), and incorporate the spatial grouping on both the second-layer inputs and outputs. This gives us a total of 768 parameter vectors. We then assign the geometric label to each segment based on the average confidence scores within the segment. We evaluate the performance by computing the accuracy of assigning the correct geometric label to a pixel.

3.5 Experiments and Results

We evaluate the proposed algorithm on two applications: (1) object recognition and detection on SUN09 dataset with 107 object categories; (2) the multi-task cascaded structure that composes scene categorization, depth estimation and geometric labeling on multiple datasets as described in Section 3.4.

Table 3.1: Performance of object recognition and detection on SUN09 dataset.

Model	Object Recognition (% AP)	Object Detection (% AP)
Chance	5.34	N/A
Baseline (w/o context)	17.9	7.06
Single model per object	22.3	8.02
Independent model	22.9	8.18
State-of-the-art [15]	25.2	8.33
θ-MRF (l_2-regularized)	**26.4**	**8.76**
θ-MRF (l_1-regularized)	**27.0**	**8.93**

Table 3.2: Performance of scene categorization, geometric labeling, and depth estimation in CCM.

Model	Scene Categorization (% AP)	Geometric Labeling (% AP)	Depth Estimation (RMSE in m)
Chance	22.5	33.3	24.6
Baseline(w/o context)	83.8	86.2	16.7
State-of-the-art [83]	86.1	88.9	15.2
CCM [53] (our implementation)	83.8	87.0	16.5
θ-MRF (l_2-regularized)	**85.7**	**88.6**	**15.3**
θ-MRF (l_1-regularized)	**86.3**	**89.2**	**15.2**

3.5.1 Overall performance on multiple tasks in CCM strcuture.

Table 3.2 shows the performance of different methods on the three tasks composed into the cascaded classification model (CCM) [53]. "Baseline" means the individual classifier for each task on the first layer, "State-of-the-art" corresponds to the state-of-the-art algorithm for each sub-task respectively for that

specic dataset, and "CCM" corresponds to the second-layer output for each subtask in the CCM structure. The results are computed as the average performance over 6-fold cross validation. With the semantic and spatial regularization, our proposed θ-MRF algorithm improves over the CCM algorithm that also uses the same set of tasks for prediction. Finally, we perform better than the state-of-the-art algorithms on two tasks and comparably for the third.

Is θ-MRF "complementary" to label-MRF? In this experiment, we also consider the MRF over labels [123] together with our θ-MRF for depth estimation. The combination results in a lower root-mean-square-error (RMSE) of 15.0m as compared to 15.2m for θ-MRF alone and 16.0m for label-MRF alone. This indicates that our method is complementary to the traditional MRF over labels.

3.5.2 Overall performance on SUN09 object detection.

Table 3.1 gives the performance of different methods on SUN09 dataset, for both object recognition (predicting the object presence) and object detection (predicting the object location).

- Baseline (w/o context): the baseline object detectors trained by [33], which are also used to generate the initial detection results used as inputs for our algorithm and the state-of-the-art algorithm.
- Single model: a single classifier is trained for each object category, not varying across different locations. In the following, if not specified, we use a l_1-regularized linear regression as the classifier.
- Independent model: this means an independent classifier is trained for the presence of an object for each region. There is no information sharing between the

models belonging to different locations of the same category, or different categories.

- State-of-the-art: This is the tree-based graphical model proposed in [15], which explicitly models the object dependencies based on labels and detector outputs.[2]
- The proposed θ-MRF algorithm, which shares the models spatially within an object category and semantically across various objects. We evaluate both the l_1 and l_2 regularization on the potentials.

Table 3.1 shows the location-specific model (Independent) is better than the general model (Single model), which confirms our intuition that the contextual model is location-specific. Furthermore, our approach that shares parameters spatially and semantically outperforms the independent model without these regularizations. We also note that our algorithm can achieve comparable performance to the state-of-the-art algorithm, without explicitly modeling the probabilistic dependency between the objects labels.

We study the relative improvement of the proposed parameter sharing algorithm over the non-parameter-sharing algorithm (Independent model in Table 3.1) on object categories with different number of training samples in the SUN09 object recognition task. The relative improvement on object categories with less than 200 training samples is 34.2%, while the improvement on objects with more than 200 training samples is 11.5%. Our parameter sharing algorithm helps the infrequent objects implicitly make use of the data of frequent objects to learn better models.

We give two examples in Fig. 3.3, focusing on two infrequent object cate-

[2]We evaluate the contextual model in [15] using the software published by the authors: `http://web.mit.edu/~myungjin/www/HContext.html` and report the average performance on multiple runs.

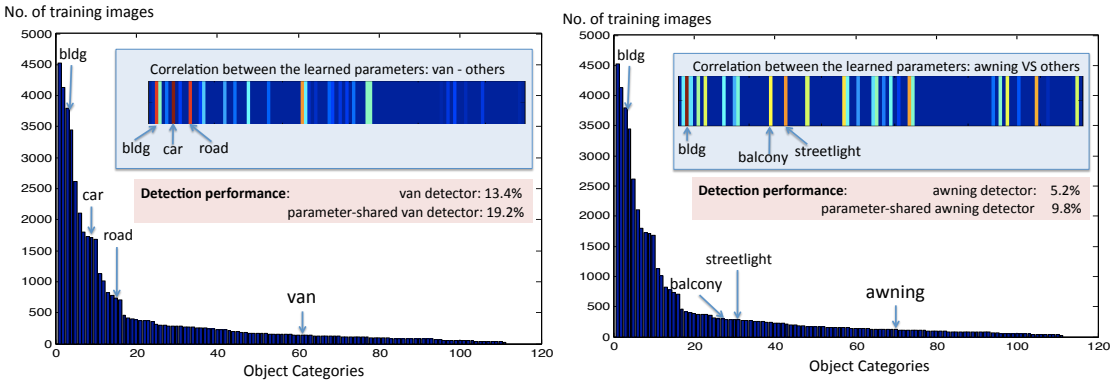


Figure 3.3: Examples showing that infrequent object categories share parameters with frequent object categories.

gories: van and awning, respectively. The histogram in the figures shows the number of training instances for each object category. The color bar shows the correlation between the learned parameter of the object with the parameters for other objects. The redder indicates the higher correlation between the parameters of the respective categories. Figure 3.3-left shows that the van category has few training instances, turn out to share the parameters strongly with the categories of car, building and road. Similarly, Figure 3.3-right shows how the learned awning parameters with other categories. We note that in the dataset, awning and streetlight are not highly co-occuring, thus initially when we create the semantic groups, these two objects *do not* appear simultaneously in any group. However, the semantic groups containing streetlight and the semantic groups containing awning both contain objects like road, building, and car. Through our θ-MRF algorithm, the sharing information can be transferred.

Effect of different priors. We compare our spatially-grouped and semantically-grouped regularization with other parameter sharing algorithms such as the prior-based algorithms in Figure 3.4.

- Global prior (Fig. 3.4-left): all the classifiers share the prior β_0, and the parameter for

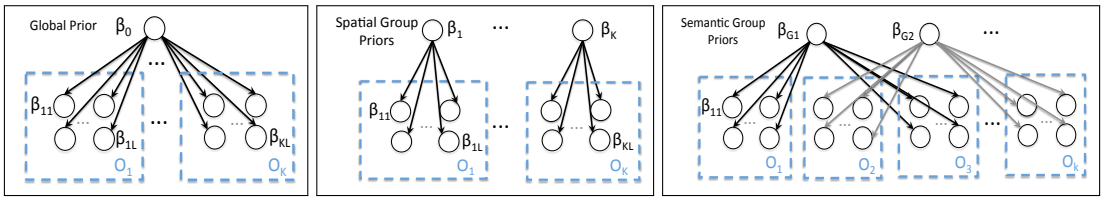


Figure 3.4: Some baseline prior-based algorithms we compare the propose algorithm with. From left to right: these models use global prior, spatial-based prior, and semantic-based prior.

Table 3.3: Results for different parameter sharing methods on object recognition task.

Models	Object Recog. (% AP)
No prior, l_2-sparsity	22.5
Global prior	22.8
Spatial priors	23.6
Spatial θ-MRF	**24.6**
Semantic priors	24.0
Semantic θ-MRF	**25.2**
Full θ-MRF	**26.4**
Full θ-MRF, l_1	**27.0**

a classifier is defined as: $\theta_{k,l} = \beta_0 + \beta_{k,l}$. Assuming zero-mean gaussian distribution for each β, we have the regularization term in Equation 3.1: $R(\mathbf{\Theta}) = \sum_{k,l} R(\theta_{k,l}) = \sum_{k,l} (\lambda_0 ||\beta_0||_2 + \lambda_{k,l} ||\beta_{k,l}||_2)$.

- Spatial prior (Fig. 3.4-middle): the classifiers for the same object O_k at different locations share a prior θ_k,i.e.,$\theta_{k,l} = \beta_k + \beta_{k,l}$. Thus we have $R(\mathbf{\Theta}) = \sum_{k,l} R(\theta_{k,l}) = \sum_{k,l} (\lambda_k ||\beta_k||_2 + \lambda_{k,l} ||\beta_{k,l}||_2)$.
- Semantic prior (Fig. 3.4-right): the classifiers from the same semantic group share a prior β_{G_i}, and the parameter for a classifier is defined as: $\theta_{k,l} = \beta_{G_i} + \beta_{k,l}$, where $\theta_{k,l} \in G_i$. Thus we have $R(\mathbf{\Theta}) = \sum_{k,l} R(\theta_{k,l}) = \sum_{k,l} (\lambda_{G_i} ||\beta_{G_i}||_2 + \lambda_{k,l} ||\beta_{k,l}||_2)$. The semantic groups are generated by an agglomerative clustering based on the co-occurence of objects.
- Spatial θ-MRF and Semantic θ-MRF (Fig. 3.1): the proposed regularizations in Section 3.3 and $l-2$ norm is used as the regularization form.

Table 3.3 shows that the proposed θ-MRF algorithms outperform the prior-based algorithms in both the spatial-grouping and semantic-grouping settings. Sharing the only global prior across all tasks performs slightly better than the independent l_2 regularization based classifier. Modeling the spatial and semantic interactions by both methods (adding priors or adding edges) improve the performance, while the θ-MRF based approach is more effective, especially with l_1 norm.

Visual grouping. Figure 3.5 illustrates the effect of our proposed parameter sharing. For each object in a location, we show the top three contextual inputs learned by our approach. In Figure 3.5-left, we show where the highest positive weights locate in order to detect shoes at different regions. We note that to detect shoes in topper part of the image, shelves, closet and box are the most important contextual inputs; while floor and wall play a more important role in detecting shoes at the bottom of the image. The results also show that the two neighboring shoe regions (row 2 and row 3) share similar context, while far-away ones (row 1) do not. This reflects our target-based spatial interaction within the parameter-MRF.

In Figure 3.5-right, we show a group of parameters (corresponding to different regions for oven, refrigerator, and sink) that share semantic edges with each other on the parameter-MRF. We note that they share the high weights given to stove at the bottom-right and microwave at the middle-left. All these objects have very few training examples. With the proposed semantic constraint, they can implicitly leverage information from each other in the training stage. Besides we also note the spatially smooth effect on the figures, which is resulted from our source-based spatial interactions.

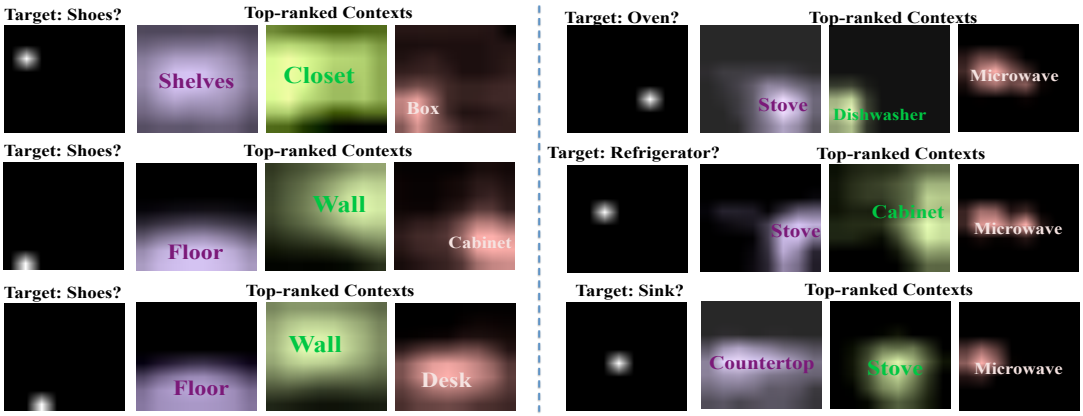


Figure 3.5: Examples of the visual context learned from the proposed algorithm. Six examples are given. In each example, the left figure illustrates the task: whether the white region belongs to the target category. The following three figures shows the contextual inputs (showing the spatial map) which have the top ranked weights (highest positive elements of the parameters) .

Conclusion. We propose a method to capture structure in the parameters by designing an *MRF over parameters*. Our evaluations show that our method performs better than the current state-of-the-art algorithms on four different tasks (that were specifically designed for the respective tasks). Note that our method is complementary to the techniques state-of-the-art methods use for the respective tasks (e.g., MRF on the labels for depth estimation), and we believe that one can get even higher performance by combining our θ-MRF technique with the respective state-of-the-art techniques.

3.6 Summary

In this chapter, we propose the θ-MRF algorithm, which models the sparse interactions between the classifier parameters based on spatial and semantic context. In this algorithm, we allow taking contextual features/attributes from all regions in the image as input, and then model the contextual dependence between

the different regions of interest. In particular, two sets of parameters are encouraged to have similar values if they are spatially close or semantically close. Our method is, in principle, complementary to other ways of capturing context such as the ones that use a graphical model over the labels instead. Experiments on various tasks and various datasets show even with a learning model as simple as regression the proposed algorithm works better than or comparably to the state of the art algorithms. This work has first appeared in the following publication: Li, Saxena, Chen [76].

CHAPTER 4

DISCOVERING ATTRIBUTES: CONTEXTUAL-META OBJECTS

In the Chapter 2 and Chapter 3, we focus on leveraging contextual information from other existing tasks. In Chapter 4 and Chapter 5, we look into another direction: introducing new contextually informative attributes and developing algorithms to automatically discover such attributes, to further improve the performance of multiple scene understanding tasks, especially object recognition and scene recognition.

4.1 Introduction

Recognizing objects and scenes is one of the central problems in computer vision. Many recent works leverage contextual information surrounding the object-of-interest for enhanced recognition [22, 43, 54, 67, 108, 113]. These typically leverage labeled data from other object categories to learn contextual relationships. This leads to two undesirable consequences.

Unlabeled regions: First, regions in the images that are unlabeled are often neglected. In most scenarios such as the popular MSRC [1] or PASCAL [26] datasets, not all regions in the images are accounted for by the manually chosen categories that are labeled. For instance, 28.18% of the pixels in MSRC and 54.74% of pixels in the PASCAL 2007 dataset are not a part of the labeled categories. See Figure 4.1. Do these unlabeled regions contain useful information that is worth capturing? We conduct human studies on recognizing objects in

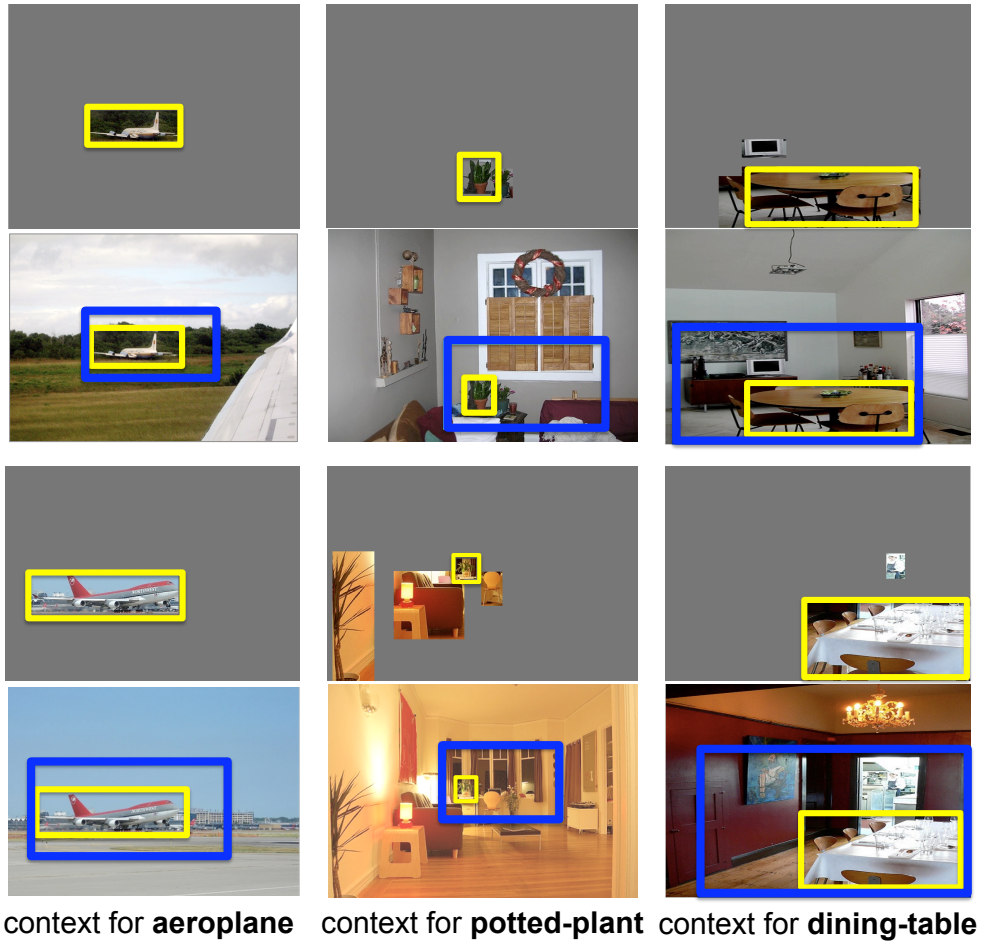


Figure 4.1: Many approaches to contextual reasoning for object detection model the relationship of the object-of-interest (yellow boxes) to other object categories labeled in the dataset. They do not leverage unlabeled regions in the images that do not belong to these manually chosen labeled categories, resulting in a highly myopic view of the scene (1^{st} and 3^{rd} row). Our approach (blue boxes) intelligently leverages information present in these unlabeled regions to better detect the object of interest. From left to right: unlabeled regions like sky and grass provide context for aeroplanes, and unlabeled objects like windows and paintings are contextually relevant for potted-plants and dining-tables respectively. The granularities of our blue boxes relative to yellow boxes are learned adaptively.

low-resolution images[1] from the PASCAL dataset. We find that subjects perform better in scenarios where they see the entire image including the unlabeled regions (Figure 4.2). This indicates that the unlabeled regions, which are often

[1]Parikh et al [108] showed that humans need contextual information for recognition only when the appearance information is impoverished, such as in low-resolution images.

Figure 4.2: Human subjects were shown images excluding the unlabeled regions (left), as well as entire images (middle). The object to be recognized is shown with and without a yellow-outline (to avoid distraction). Our results (right) indicate that subjects can recognize objects significantly more reliably if information from the unlabeled regions is available. These experiments were conducted on 394 PASCAL 2007 images containing 897 objects from 20 categories.

discarded, indeed contain useful contextual information.

Adaptive granularity: Second, in focusing only on labeled regions in the image, models inadvertently limit the contextual information captured to the granularity implicit in the labels. For instance, most works exploring contextual models for the PASCAL dataset consider only object-level information, such as co-occurrence, relative location and relative size [22, 33]. We argue that the granularity at which contextual information is most helpful is category specific. While an "aeroplane" may only need to consider a neighboring sky region as context, "dining-table" can benefit from the entire scene layout. Moreover, while a "sheep" may be surrounded by relevant contextual information all around it, the most reliable contextual information for a "bicycle" is the person on top. Therefore, it is crucial that context is extracted from regions that adapt to different objects.

In this part of work, we overcome both these drawbacks and propose a contextual cue that exploits unlabeled regions in images and automatically adapts

to each object category.

Formulation: What kind of information is contextually most useful? Information that is relevant i.e. has consistent spatial location with respect to the object-of-interest, and information that is reliable i.e. has consistent appearance across images for reliable detection. Interestingly, these are also aspects that make object models effective: capturing spatially coherent and visually consistent object-parts. Instantiating the popular hierarchical view of scenes [102, 109, 129] (parts are to objects as objects are to scenes), we can cast the problem of learning relevant contextual regions in a scene as that of learning detectors for "objects", which we call contextual meta-objects (CMOs). These CMOs form our proposed contextual cues. *Any* existing object detector can be used to learn our CMOs. In fact, the detector that one trains to detect objects-of-interest (OOIs) can be seamlessly (and conveniently so!) used to learn CMOs, essentially boosting the performance of the OOI detector without designing complex algorithms or cumbersome learning procedures.

Summary of approach: How can we extract meaningful contextual information from unlabeled regions? Our approach exploits the following key observation: object bounding boxes do not simply provide us with information about what the object looks like which can be used to train a detector for the object. The presence of an object at a certain location in a natural image also provides an anchor point that suggests a meaningful alignment among the scenes containing these objects. Given images labeled with bounding-boxes of the OOI, we align and cluster the images such that each cluster contains images with similar contextual information surrounding the OOI. We identify a CMO region around the OOI that best captures this contextual information. Note that the

granularity at which the CMO region is defined is not fixed, and adapts to the content of the images. Any off-the-shelf object detector can then be trained to detect our CMOs. During testing, we apply the CMO detector on the test image, and the score of the detection captures our contextual cue. This cue can be combined with the OOI detection score, or other contextual cues for enhanced OOI detection.

Contributions: In this chapter, we effectively exploit the unlabeled regions in images that are often neglected by most existing works, to extract contextually relevant cues for enhanced object detection. Our contributions are three-fold. First, we discover contextual regions that automatically adapt to the object category of interest in order to capture contextual interactions at varying granularities (the entire scene, inter-object, even intra-object) for different categories. Second, we cast the problem of extracting contextual regions in scenes into the problem of learning object models. This allows us to employ any off-the-shelf object detector to learn our contextual cue; a convenient choice being the object-of-interest detector whose performance we hope to enhance via contextual reasoning. This simplicity makes our proposed cue easily accessible to the community. Lastly, our approach achieves higher detection performance when using a *single* labeled category, than the state-of-the-art approach [33] that utilizes labels for *all* 20 object categories on the PASCAL 2007 dataset. This demonstrates our effective use of unlabeled regions. We use our approach to boost performance of several object detectors, and show that our contextual cue compares favorably to, and complements other sources of context. Our adaptive selection of the granularity of contextual information outperforms the fixed-granularity counterpart.

4.2 Related Work

Many recent works leverage contextual information for enhanced recognition and localization of objects in natural images. Various sources of context have been explored, ranging from the global scene layout, interactions between objects and regions, as well as local features. Divvala et al [24] survey and study the effectiveness of different contextual cues and combine them to achieve superior performance. We view our novel cue that extracts useful contextual information from unlabeled regions as complementary to existing cues, and can be easily integrated in most contextual models.

Fixed-granularity models: Many existing works commit to a fixed granularity of contextual information. To incorporate scene-level information, Torralba et al. [137] use the statistics of low-level features across the entire scene to prime object detection. Hoiem et al. [57] use 3D scene information to provide priors on potential object locations. Park et al [110] use the ground plane estimation as contextual information for pedestrian detection. Probabilistic models have been proposed to capture the local interactions between neighboring regions [54, 67, 94], objects [22, 33, 108, 113, 148], or both [7, 41]. Sadeghi et al [119] propose and detect visual phrases which correspond to chunks of meaning bigger than objects and smaller than scenes. While the visual phrases in [119] are manually labeled in the dataset, our work can be thought of as learning the visual phrases in an unsupervised way. Our learned composites, however, may not have a clearly defined semantic meaning. While most works focus on one level of interaction, Galleguillos et al [41] explore contextual interactions at multiple levels. However, this multi-level aspect is gained by explicitly using different

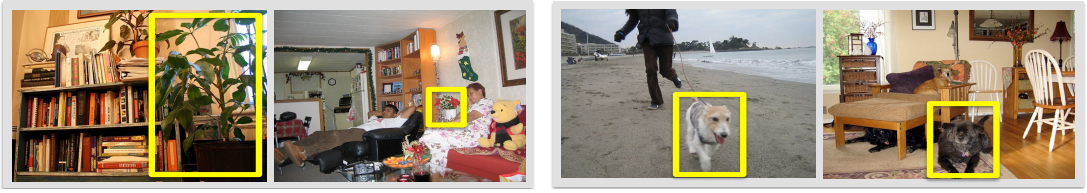
models for each interaction level. In contrast, our approach allows for adaptively picking different granularities of contextual information within the same framework.

Leveraging unlabeled information: A natural way to incorporate information from unlabeled regions in images is to build a global descriptor for the entire image to provide contextual cues. Though global image statistics show great potential in priming object detection [137], very modest improvement can be achieved when applied to datasets like PASCAL (also confirmed in our experiments), where images have poor alignments due to the high variance in object scales, poses, etc. [15]. Instead, local neighboring regions tend to be more effective. Wolf et al [146] sample contextual information from pre-defined relative locations. Dalal et al [18] show that simply increasing the size of the person bounding box by a small amount boosts the accuracy of pedestrian detection. This is equivalent to leveraging potentially unlabeled regions in close proximity of the object-of-interest. Our approach instead automatically and adaptively determines the extent of contextual information to be captured around different object categories. Felzenszwalb et al [33] often detect parts that lie slightly outside the sliding window, and use these detections to refine their bounding box prediction. Lee et al [74] utilize a few labeled categories to discover other object categories in the 'background' that have consistent appearance and are contextually coherent with the labeled categories. While similar in philosophy to our work, our solution is quite different, as is the problem setting of enhancing object detection.

4.3 Approach

In order to extract contextual cues that exploit unlabeled regions in images, we work in the most extreme setting where only one object category is labeled (with bounding-boxes) in the training dataset, leaving a large portion of the images unlabeled. We can seamlessly incorporate more labeled categories in our approach, as we describe later.

Our goal then is to extract useful contextual regions, given images labeled with ground-truth bounding-boxes for just one object category i.e. the object-of-interest (OOI). The ground-truth OOI may occupy different proportions of images across the dataset, as seen in Figure 4.3(a). Moreover, images may exhibit different contextual settings, as seen in Figure 4.3(b). To better model these, we first group the training images that exhibit similar contextual extent and content (Sections 4.3.1 and 4.3.2). As stated earlier, we cast our problem of extracting a contextual region in a scene as that of learning an object model, which we call contextual meta-object (CMO), for which any off-the-shelf detector can be used. Each cluster of images is used to train a different *component* of our CMO detector (Section 4.3.3). During testing, we run both the trained OOI and CMO detectors, and the score of the latter provides contextual information for the former. While any contextual reasoning model can be used to integrate the two, in our implementation we adopt the simple contextual re-scoring scheme from [33] (Section 4.3.4).



(a) (b)

Figure 4.3: Context around objects (potted-plant and dog shown in yellow boxes) can vary in extent (left) as well as content (right).

4.3.1 Contextual-extent-based Clustering

We consider the contextual-extent of an image to be the portion of the image that lies outside of and surrounds the OOI. Intuitively, we wish to group all images that contain the OOI with similar poses, at similar scales, and in similar locations with respect to the rest of the scene.

We use all the training images, as well as their left-right flipped versions. To enhance consistency in the data, we first divide the images into 2 groups: one containing images with the OOI ground-truth bounding boxes on the right, and the other containing images with the OOI on the left. We then employ the following procedure for both groups. Consider an image I^i consisting of n_B ground-truth OOI bounding-boxes $\{B_j\}$, $j \in \{1, \ldots, n_B\}$. I^i is described by a set of n_B five-dimensional descriptors $F^i = \{f_1, \ldots, f_j, \ldots, f_{n_B}\}$

$$f_j = \left[\frac{\left|x^i_{tl} - \bar{x}^i_j\right|}{s^i_j} \; \frac{\left|y^i_{tl} - \bar{y}^i_j\right|}{s^i_j} \; \frac{\left|x^i_{br} - \bar{x}^i_j\right|}{s^i_j} \; \frac{\left|y^i_{br} - \bar{y}^i_j\right|}{s^i_j} \; \frac{h^i_j}{w^i_j} \right] \tag{4.1}$$

where $\left(x^i_{tl}, y^i_{tl}\right)$ and $\left(x^i_{br}, y^i_{br}\right)$ are the co-ordinates of the top-left and bottom-right corners of the image I^i respectively, $\left(\bar{x}^i_j, \bar{y}^i_j\right)$ are the co-ordinates of the center of the j^{th} ground-truth OOI bounding-box in image I^i, and s^i_j, h^i_j, and w^i_j are the scale, height and width of the j^{th} OOI bounding-box respectively. This descriptor captures the extent of the contextual information in the image (in terms of relative location and scale) with respect to the OOI present in the image, as

well as the aspect ratio of the OOI. For each of the groups mentioned above, we cluster the descriptors $\cup_{i=1}^{N} F^i$ collected from all N training images containing the OOI into K clusters using k-means clustering. This results in a total of $2K$ context-extent based clusters of images. In our implementation, we use $K = 3$.

For all images in the k^{th} cluster, $k \in \{1, \ldots, K\}$, we determine the extent of the CMO to be the largest bounding-box surrounding the OOI such that the CMO is entirely contained within at least 80% of the images in the cluster. This bounding-box indicates the presence and extent of the CMO we will learn via an off-the-shelf object detector. We note that since the extent of the CMO is not tied to the OOI bounding-box itself, and instead depends on the layout of the OOI in the scene, it can freely capture any relevant information in the image, including unlabeled regions, at any granularity. In fact, CMOs corresponding to the different clusters capture contextual information at different granularities for the same object category. We note that a training image I^i containing n_B ground-truth OOI instances will have n_B corresponding instances of the CMO for training.

4.3.2 Contextual-content-based Clustering

While we ensure that the extent of context is similar among the images within the $2K$ clusters, the contextual setting or content of the images could be quite varied (Figure 4.3(b)). We further cluster each of the $2K$ groups based on their content. We extract gist [136] features within the CMO box, and perform k-means clustering on these features to divide each of the $2K$ groups into two clusters. In our implementation, when less than 40 images are assigned to a

cluster, we drop the cluster and re-assign the corresponding images to the remaining clusters. Therefore, we have $M \leq 4K$ clusters, which varies with categories. For the PASCAL VOC 2007 dataset, we find a total of 198 clusters across the 20 categories. These clusters now have CMOs defined that are consistent in appearance as well as spatial relationships with respect to the OOI, and thus have the potential to provide useful contextual information to enhance the OOI detection.

4.3.3 CMO Detection

We now describe how we use the above clusters to learn our CMO detector. As stated earlier, we can use any off-the-shelf object detector, which we treat as a black-box parameterized by a model θ, learnt via a training procedure f_{train} that takes in training data of the form $(\mathcal{P}, \mathcal{N})$. $\mathcal{P} = \{(I_1^+, B_1), \ldots, (I_k^+, B_k)\}$ is a set of positive image and bounding-box pairs, and $\mathcal{N} = \{(I_1^-), \ldots, (I_l^-)\}$ is a set of negative images. The training procedure can be viewed as

$$\hat{\theta} = \text{optimize } f_{\text{train}}(\mathcal{P}, \mathcal{N}; \theta). \tag{4.2}$$

The detector can then be evaluated on a test image I, via the inference procedure f_{infer} to obtain a detection (B, s) including a bounding box B and a score s

$$(B, s) = f_{\text{infer}}\left(I; \hat{\theta}\right). \tag{4.3}$$

We see that the above formulation holds for a variety of detectors, be it sliding-window based [18, 33, 143] or hough-transform based [75]. So how do we use one of these black-box detectors to train our CMO detector using the M clusters formed in Section 4.3.2? In order to not commit to the hard-clustering

which was blind to the choice of detector that would follow, we train an M-component detector using an EM style approach, similar to the strategy employed in [33]. We initialize the components using the above clustering, and train M detectors to obtain $\hat{\theta}^m$ using $f_{\text{train}}(\mathcal{P}^m, \mathcal{N}^m; \theta)$, where $\mathcal{P}^m$ is the set of images in the m^{th} cluster and the CMO windows contained (Section 4.3.1), and $\mathcal{N}^m$ are negative images. We then infer these M components on all positive training images (across all M clusters), and re-assign each "ground-truth" CMO box B (Section 4.3.1) in the training set to the component that best explains it:

$$\tilde{m} = \underset{m \in \{1,\ldots,M\}}{\text{argmax}} \left(s^m - t^m\right), \tag{4.4}$$

$$s.t.(B^m, s^m) = f_{\text{infer}}\left(I; \hat{\theta}^m\right), \text{overlap}(B^m, B) > 0.5 \tag{4.5}$$

where t^m is a bias term used to normalize the scores across the components, and is set to the 5^{th} percentile of the scores assigned by component m to all detections in the training images. Each component m is now re-trained, but with the new set of positive examples, to obtain an updated $\hat{\theta}^m$, and the iterations continue. We find that 3-5 iterations suffice. During testing, all M components are evaluated on the test image. Non-maximal suppression is used on the resultant detections to eliminate repeated detections.

4.3.4 Contextual Re-scoring

We now describe how we use our CMO to provide context to the OOI detection. We evaluate our CMO as well as the OOI detectors on the test image. The presence as well as the location of the CMO can provide useful contextual in-

formation. However in this work, we only consider the presence i.e. the CMO detection score. Based on the CMO detections, we wish to re-score the detected OOI bounding boxes. While any contextual reasoning mechanism can be used to this end (such as [24, 33, 43, 114]), we train a classifier, similar to Felzenszwalb et al [33]. Let D be the set of detections obtained for the OOI detector. Each detection (B, s), $(B, s) \in D$ is formed of a bounding-box with co-ordinates $B = (x_{tl}, y_{tl}, x_{br}, y_{br})$ (overloaded from previous sections) and a score s. Let s_C be the score of the highest scoring CMO detection (across all M components) found in the image. To re-score an OOI detection (B, s), we build a 6-dimensional descriptor consisting of the original score of the OOI detection, the top-left and bottom-right bounding box coordinates normalized by image size, and the contextual information provided by the CMO detection, captured via s_C:

$$\psi_{1c} = [\sigma(s)\ x_{tl}\ y_{tl}\ x_{br}\ y_{br}\ \sigma(s_C)] \tag{4.6}$$

where $\sigma(x) = 1/(1 + exp(-\alpha x))$, $\alpha = 1.5$, as in [32].

This ψ_{1c} descriptor is fed to a classifier h trained to separate correct OOI detections from false positives. The OOI bounding-box B is assigned a new score $\tilde{s} = h(\psi_{1c})$. In our implementation, the classifier h is an SVM with a polynomial kernel (parameters set via cross-validation), similar to [32, 33]. The training data are obtained by running the trained OOI detector on labeled training data, and collecting correct detections and false positives. We note that ψ_{1c} uses labeled data from only one object category, and still captures contextual information.

Our approach is not restricted to using labels from only one object category. If more object categories are available, they can be seamlessly incorporated. A CMO detector would be trained for each labeled object category. During test

time, let $(D_1, \ldots, D_n)$ be the set of OOI detections obtained for n different object categories (for PASCAL $n = 20$ if all categories are considered). Let $(s_{C_1}, \ldots, s_{C_n})$ be the scores of the highest scoring CMO detections for each of the corresponding n CMO detectors. The contextual descriptor to re-score an OOI detection $(B, s) \in D_i$ is now $n + 5$ dimensional

$$\psi_{nc} = [\sigma(s)\ x_{tl}\ y_{tl}\ x_{br}\ y_{br}\ \sigma(s_{C_1}) \ldots \sigma(s_{C_n})]\,. \tag{4.7}$$

Similar to traditional approaches that exploit context, Felzenszwalb et al [33] only use other OOI object categories to provide context, via a descriptor

$$\psi_{no} = [\sigma(s)\ x_{tl}\ y_{tl}\ x_{br}\ y_{br}\ \sigma(s_{D_1}) \ldots \sigma(s_{D_n})] \tag{4.8}$$

where s_{D_i} is the score of the highest-scoring OOI detection from the i^{th} OOI category. We note that this descriptor is identical to the one used in [33], which we compare to in our experiments.

Finally, a contextual descriptor capturing contextual information provided by all n CMO and OOI detectors is given as

$$\psi_{nco} = [\sigma(s)\ x_{tl}\ y_{tl}\ x_{br}\ y_{br}\ \gamma(C)\ \gamma(D)] \tag{4.9}$$

$$\gamma(C) = [\sigma(s_{C_1}) \ldots \sigma(s_{C_n})] \tag{4.10}$$

$$\gamma(D) = [\sigma(s_{D_1}) \ldots \sigma(s_{D_n})]. \tag{4.11}$$

A special case of Equation 4.9 for $n = 1$ differs from Equation 4.6 by one-dimension corresponding to $\sigma(s_D)$, the score corresponding to the highest-

scoring OOI detection. Since the use of $\sigma(s_D)$ does not require additional training data, and is obtained by using labeled data from a single object-category, we replace Equation 4.6 with the following in our experiments.

$$\psi_{1co} = [\sigma(s)\ x_{tl}\ y_{tl}\ x_{br}\ y_{br}\ \sigma(s_C)\ \sigma(s_D)] \tag{4.12}$$

4.3.5 Computational Efficiency

Our approach of discovering the CMO regions mainly contains two stages of K-means clustering. The complexity of K-means clustering is $O(tnkd)$, where t is the number of iterations, n is the number of data instances, k is the number of clusters, and d is the feature dimension of each data instance. Given the limited number of positive instances, the small number of clusters, and the relatively low-dimension features we consider, the two stages of clustering is quite efficient. In the training stage, after discovering the CMO bounding boxes, we simply adopt any off-the-shelf object detection techniques to train the CMO detectors. The training of the re-scoring classifier uses the SVM classifier and the dimension of the input features is linear to the number of object categories (which is usually less than hundreds in most existing datasets). In the testing stage, the complexity mainly depends on the original inference algorithm of the detectors being adopted, besides which a small overhead is added by the inference of the re-scoring classifier.

4.4 Experiments and Results

We evaluate our approach using the PASCAL VOC 2007[2] challenge dataset and protocol [26], which contains 9963 images of realistic scenes, containing ground-truth bounding boxes for 20 object categories. While we provide results with other object detectors in subsequent experiments, we first perform several comparisons and analyses using the publicly available implementation [32] of the state-of-the-art deformable parts-based object detector [33].

4.4.1 Quantitative results

We first provide several quantitative evaluations, followed by some qualitative illustrations.

Useful contextual information from unlabeled regions: We first evaluate whether our proposed cue captures useful contextual information extracted from the unlabeled regions. We compare the performance of the baseline OOI detector to the contextually re-scored detector, using our proposed cue as the source of contextual information as described in Equation 4.12. The results can be seen in Table 4.1 (Base w/o context vs. CMO). We see that across all 20 categories, our method outperforms the state-of-the-art OOI detector, indicating that our contextual cue does in fact extract useful contextual information from unlabeled regions.

[2]We are not proposing a novel object detector. Instead, we will be performing a variety of comparisons to demonstrate the effectiveness of our proposed contextual cue. As recommended by the challenge organizers [26], we work with the 2007 test-set, which is the latest PASCAL test-set with publicly available annotations.

	CMO Ratio
OOI	37.2%
Other Categories	17.5%
Unlabeled Regions	45.3%

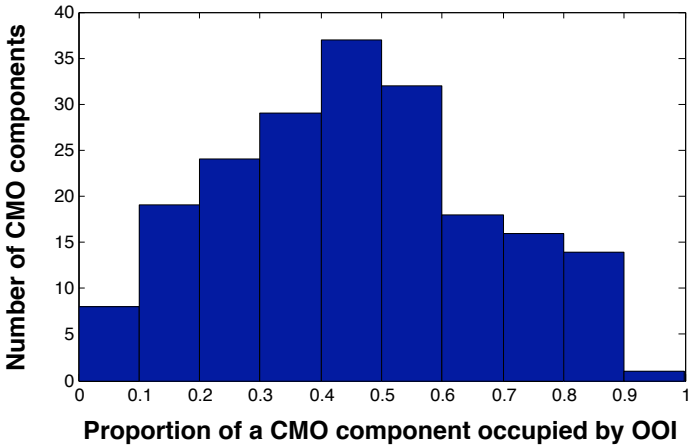


Figure 4.4: **Left**: The proportion of highest-scoring CMO detections on positive testing images occupied by different content. **Right**: The average proportion of a CMO component detection occupied by OOI. The truly multi-granular nature of our learnt contextual cues is evident.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbik	pers	plant	sheep	sofa	train	tv	MEAN
Base [33]	28.9	59.5	10.0	15.2	25.5	49.6	57.9	19.3	22.4	25.2	23.3	11.1	56.8	48.7	41.9	12.2	17.8	33.6	45.1	41.6	32.3
Scene (~ [137])	**30.9**	56.6	**11.5**	**18.5**	23.1	49.1	58.1	21.0	23.1	23.9	25.1	12.3	**59.9**	47.7	42.1	12.3	19.1	33.5	45.4	40.7	32.7
EXO (~ [18])	30.2	59.6	11.0	16.5	25.1	49.6	58.7	21.2	23.2	26.1	25.3	12.0	59.7	49.0	42.7	12.4	19.8	36.9	46.0	42.7	33.4
CMO	30.5	**60.1**	11.2	17.0	**26.7**	**49.7**	**59.1**	**23.3**	**23.4**	**26.9**	**29.3**	**13.2**	59.7	**49.3**	**43.0**	**13.4**	**20.4**	**37.8**	**46.8**	**43.3**	**34.2**

Table 4.1: Average precision (AP) for all 20 categories in PASCAL VOC 2007 and mean AP across 20 categories. All methods listed use labels from only one object category. (~[·]) means the method is similar in spirit to the reference work.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbik	pers	plant	sheep	sofa	train	tv	MEAN
20OOI ([33])	31.2	61.5	11.9	17.4	27.0	49.1	59.6	23.1	23.0	26.3	24.9	12.9	60.1	51.0	43.2	13.4	18.8	36.2	**49.1**	43.0	34.1
20CMO+20OOI	**31.5**	**61.8**	**12.4**	**18.1**	**27.7**	**51.5**	**59.8**	**24.8**	**23.7**	**27.2**	**30.7**	**13.7**	**60.5**	**51.1**	**43.6**	**14.2**	**19.6**	**38.5**	**49.1**	**44.3**	**35.2**

Table 4.2: Average precision (AP) for all 20 categories in PASCAL VOC 2007 and mean AP across 20 categories in Table 4.1). All methods listed use labels from all 20 categories.

Further, in Figure 4.4 (left) we show the average proportion of the CMO detections occupied by different contents (OOI, other labeled categories, and unlabeled). We see that almost half of the CMO detections cover unlabeled regions. We also see that about 1/5 of the CMO detections capture other labeled categories in the images, even though our contextual cue does not use these

labels to explicitly learn contextual relationships between the OOI and other categories.

Adaptive scaling helps: We now compare our contextual cue to two cues that also exploit the unlabeled regions but at a fixed granularity. The first cue ("Scene") captures the entire scene, and thus operates at the global granularity, similar to the work of Torralba et al. [137]. We train a binary RBF-kernel SVM classifier on the gist descriptors [136] extracted from the entire images to discriminate between images with and without the object-of-interest. We then use the same re-scoring scheme in Equation 4.12, by replacing the $\sigma(s_C)$ with $\sigma(s_g)$, where s_g is the score of the gist-based classifier for the test image. The second cue ("EXO") is local in nature. We expand the OOI bounding-box by a fixed amount (similar to [18]) of 20% in all four directions. Instead of training our CMO on the co-ordinates determined in Section 4.3.1, we use this expanded OOI bounding-box to learn a contextual "object" which we call EXO. We use the same re-scoring scheme in Equation 4.12, by replacing $\sigma(s_D)$ with $\sigma(s_E)$, where s_E is the highest score of the EXO detections. Note all three approaches only require labels from a single object category. Table 4.1 shows that our approach CMO outperforms both fixed-granularity methods. This demonstrates our ability to effectively determine the granularity of useful contextual interactions.

Further, Figure 4.4 (right) shows the distribution of the average proportion of CMO component detections occupied by OOI. High values (right of the histogram) correspond to contextual cues that capture local context around the OOI, while low values (left of the histogram) correspond to models that capture scene level context with respect to a relatively small OOI. The large variance in the distribution demonstrates the truly *multi-granular* nature of the contextual

# of labels for training	fusion methods	mean AP
1	Scene+EXO	33.6
	CMO	34.2
	Scene+EXO+CMO	34.4
20	20OOI	34.1
	20OOI+20CMO	35.2
	20OOI+Scene+EXO	34.6
	20OOI+20CMO+Scene+EXO	35.3

Table 4.3: The mean AP across the 20 categories in PASCAL VOC 2007 for fusing various sources of contextual information.

information learnt.

Complementary cue: We now test the ability of our proposed cue to provide complementary contextual information. We consider several different sources of context popularly explored in literature: the global scene-level context ("Scene") and local context ("EXO") described above, as well as object-level context provided by other labeled objects in the images, be it a subset of the categories (nOOI) or all 20 (20OOI). In Table 4.1 we saw that CMO performs better than the global and local context individually. Table 4.3 shows that our proposed cue learnt from a *single* labeled category is comparable to (slightly outperforms) the contextual information provided by *all* 20 labeled object categoires (20OOI). We note that 20OOI corresponds exactly to the contextual approach used by Felzenszwalb et al [33]. Hence, we see that our contextual cue performs better than any of the individual sources of context, including ones that utilize significantly more amounts of labels.

Similar to [24], we analyze the performance of fusing various sources of contextual information. We use the same re-scoring method as described in Section 4.3.4 for combining the different cues, since we wish to evaluate the information captured by the cues, and not particular contextual reasoning techniques. Table 4.3 shows results of fusing different combinations of the above mentioned contextual cues. We append the highest score for each contextual

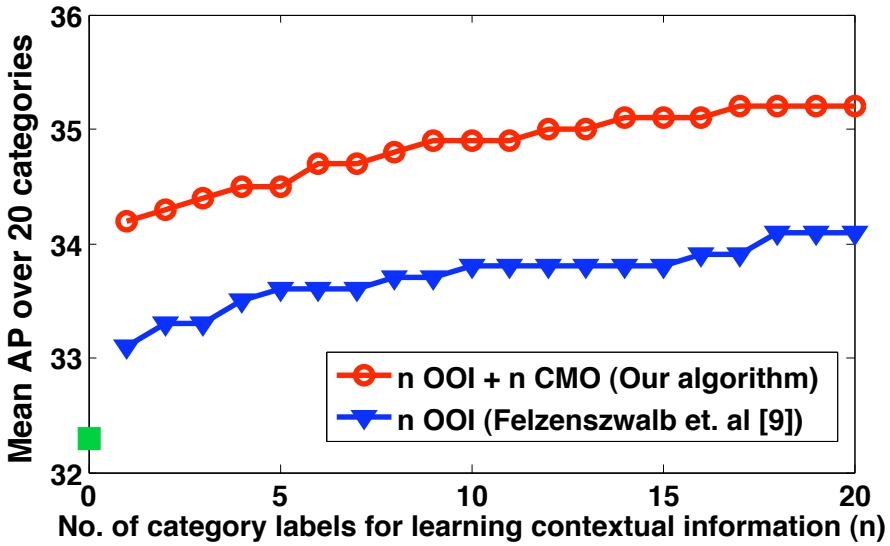


Figure 4.5: The effect of number of labeled categories on the average AP (across 20 categories).

cue being fused to re-score the OOI detection (similar to Equation 4.9). We see that our proposed cue individually performs better than the combination of *both* global and local context (Scene+EXO). Moreover, fusing our cue to these existing sources of context provides a further boost in performance, demonstrating the truly complementary nature of our cue that extracts contextual information from unlabeled regions at adaptive granularities.

We now compare our approach that leverages unlabeled regions to learn contextual cues (nOOI + nCMO, Equation 4.9), to the baseline approach of [33] (nOOI, Equation 4.8) that only utilizes other labeled categories, as the number of labeled categories is varied. For each object category (OOI), we pick n categories with highest mutual information (based on co-occurrence of categories with the OOI category across training data) to provide contextual information. Figure 4.5 shows the trends. The green point at $n = 0$ gives the mean AP of the OOI detector using no context. We can see that across the board, our approach leads to better AP than [33] while using the same amount of labeled data, demonstrating

our effective use of unlabeled regions.

A break-down of the average AP across categories when using all 20 labeled categories for both methods can be seen in Table 4.2. We see that incorporating our contextual cue that leverages unlabeled information in addition to the 20 labeled object categories (20OOI+20CMO) provides improvements over 20OOI ([33]) in 19 out of 20 categories and matches the remaining category. We see that our contextual cue on average provides a relative improvement of 12.05% over the state-of-the-art detector. We observe relative improvements of more than 20% in some categories such as bird, cat, dining-table, and dog.

Other detectors: As mentioned in Section 3, our proposed contextual cue can be extracted via any object detector, and can inturn be used to enhance the performance of the detector for the OOI. We demonstrate that here. In addition to the sliding window part-based deformable model (Parts-based Model) by Felzenszwalb et al [33] that we use in the above experiments, we consider two other popular detectors: Implicit Shape Model (ISM) [75] which is hough-transform based, and the HOG-SVM detector [18] (also sliding window). These are used as the black-box modules to train our CMO detectors, using the procedure described in Section 3.3. Table 4.4 shows the mean AP across all 20 categories in the PASCAL 2007 dataset, achieved by our implementation of these detectors, with and without the context provided by CMO. The results show that our proposed contextual cue CMO can be learnt via any detector, and consistently improves the object detection performance. Note that we use the same detectors for both OOI and CMO, indicating that no additional techniques are required for achieving these performance gains.

	OOI (mean AP)	OOI+CMO (mean AP)	Gain
ISM [75]*	12.6	15.6	23.8%
HOG-SVM [18]*	24.0	26.8	11.2%
Part-based Model [33]	32.3	34.2	8.4%

Table 4.4: Detection results with and without the proposed CMO as additional contextual information, by using different black-box detectors. $[\cdot]^*$ indicates it is our implementation of the reference work.

4.4.2 Qualitative results

Figure 4.6 shows some example CMO detections using the deformable parts-based model [33] as the detector. As quantitatively demonstrated earlier, we see that the CMO bounding-boxes contain a lot of unlabeled regions that are not labeled in the dataset, such as the sky region for aeroplanes, road for bicycle, coffee table and wall paintings for sofa, windows for potted-plants, etc. Although our approach uses labeled data only from one object category to learn the contextual cues, we learn meaningful relationships among objects as well as object parts. For example, the CMO detections for bicycle in the 2^{nd} column of Figure 4.6 consistently include a person's body as 'parts' of the CMO. In the 5^{th} and 6^{th} columns, we show detections of two CMO components corresponding to the person category. We see that both components seem to capture two people, but while the left-column models two people further apart from each other, the right-column detects two people close together. We also see intra-object CMO for cat in the last column. The false-positives shown in Figure 4.6 clearly demonstrate that our learnt CMO models fire at contextually relevant regions in images. Finally, as seen in Figure 4.4, these examples qualitatively demonstrate that our cues can adaptively learn contextual information at different granularities in the scenes.

Encouraged by the meaningful spatial interactions observed in Figure 4.6 (especially for the person category), we elaborate on that aspect a little further.

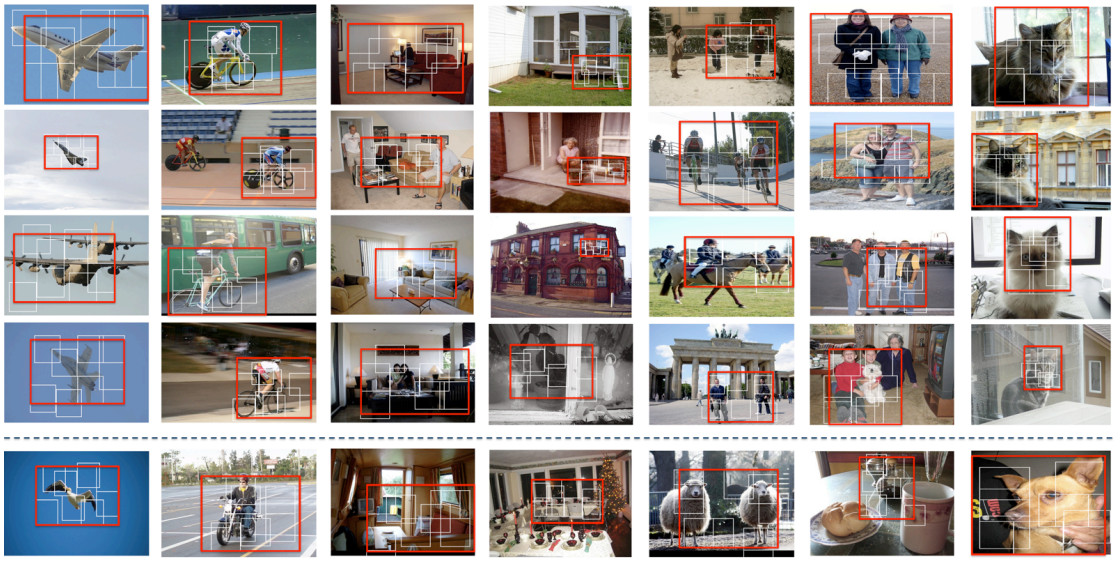


Figure 4.6: Examples of the detected contextual meta-objects (CMO). Each column shows the CMO detections for a specific category. From left to right: aeroplane, bicycle, sofa, potted-plant, person, person, cat. Each image shows the highest scoring CMO detection. Red box indicates the CMO bounding-box, while the white boxes represent the region-filters within the CMO as learnt by the deformable parts-based model [33]. The first four rows show true-positive detections, and the last row shows false-positive detections. Please refer to the authors' webpages for more results.

As we demonstrated earlier, our learnt CMO models often include objects from other labeled categories (e.g. the bicycle CMO often includes a person, a person CMO often includes another person, etc.). By examining the CMO detections on training images, we can learn a distribution of the location of each labeled category in the dataset relative to our CMO models. These spatial distributions can now be used as a prior to better guide the detection of an OOI. We show a few examples of these spatial maps in Figure 4.7. We also display the HOG feature visualizations for the corresponding CMO. While in this work we only leverage CMOs to provide co-occurrence based context, exploring the potential of our models to capture scene configurations and provide explicit spatial priming for localizing objects is part of future work.

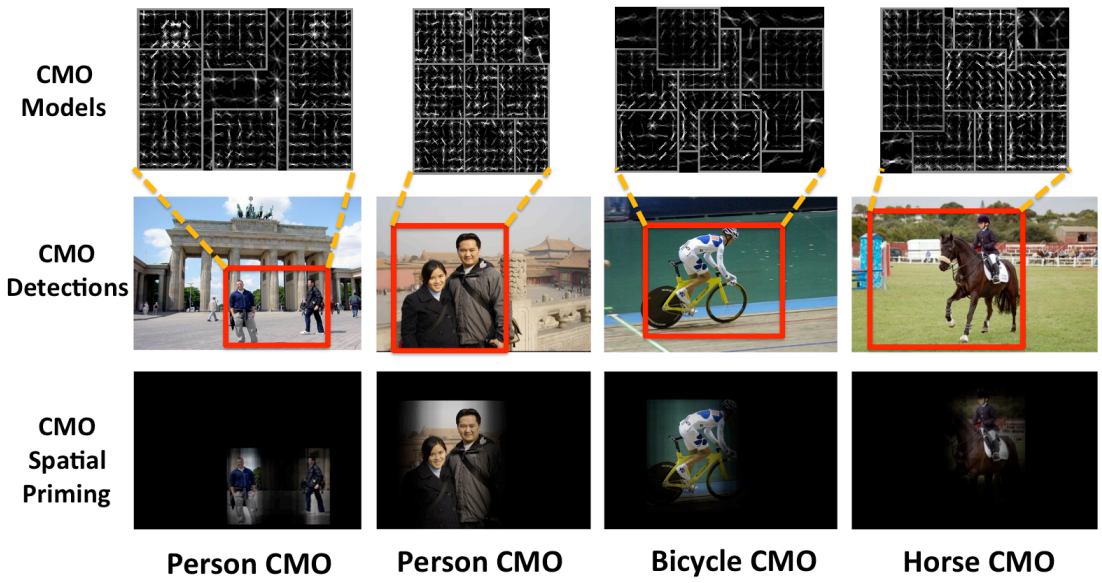


Figure 4.7: The spatial maps indicate the likelihood of a person being present given the CMO detections (red boxes). We see that the CMO provides strong priming for locations of OOIs.

4.5 Summary

In this chapter, we exploit these unlabeled regions to extract adaptive contextual cues for enhanced object detection. We utilize the labeled object bounding-box as an anchor to align scenes and learn spatially consistent and visually identifiable contextual regions. The granularity of these regions is adaptively and automatically tuned for different categories, and capture scene-level, inter-object as well as intra-object interactions. We cast the problem of learning our proposed contextual cue into that of learning object models. This allows us to utilize any off-the-shelf object detector to learn our proposed "contextual meta-objects". We present convincing quantitative and qualitative results on the challenging PASCAL VOC 2007 dataset, where we improve on the performance of several object detectors, and compare favorably to existing sources of context. The benefits of the adaptive granularity at which we extract context, and the potential of our cue to provide complementary information in addition to existing cues

are also demonstrated. These improvements do not rely on advanced modeling techniques or learning algorithms, and intelligently leverage existing technology, making them widely accessible. This work has first appeared in the following publication: Li, Parikh, Chen [87].

CHAPTER 5

DISCOVERING ATTRIBUTES: GROUPS OF OBJECTS

In Chapter 4, we promote a new attribute called contextual-meta objects (CMO), which are automatically discovered contextual regions with adaptive granularity for each object category. In this chapter, we consider another new attribute, which aims to capture the arbitrary high-order interactions between co-occurring objects in the image dataset.

5.1 Introduction

If we were to describe the image shown in Figure 5.1(a), we would perhaps say it is "an outdoor seating area with three sets of picnic-umbrella, table and chairs". Note that this description demonstrates a natural grouping of objects in the scene. This is in contrast with existing trends in computer vision of treating individual objects (or the entire scene as a whole) as the basic unit of semantics. This is not natural: it is unlikely that we would describe the scene as having "three picnic-umbrellas, three tables and nine chairs". This is because objects in scenes interact with each other in complex ways, and arguably, these interactions are what tell the story of the scene. These interactions may be of various forms such as spatial relationships, physical support, actions being performed by a subject on an object, etc. But the key observation is that all these interactions manifest themselves as predictable visual patterns in the image. While characterizing these different interactions would be valuable, simply discovering and detecting these structured visual patterns themselves is an important

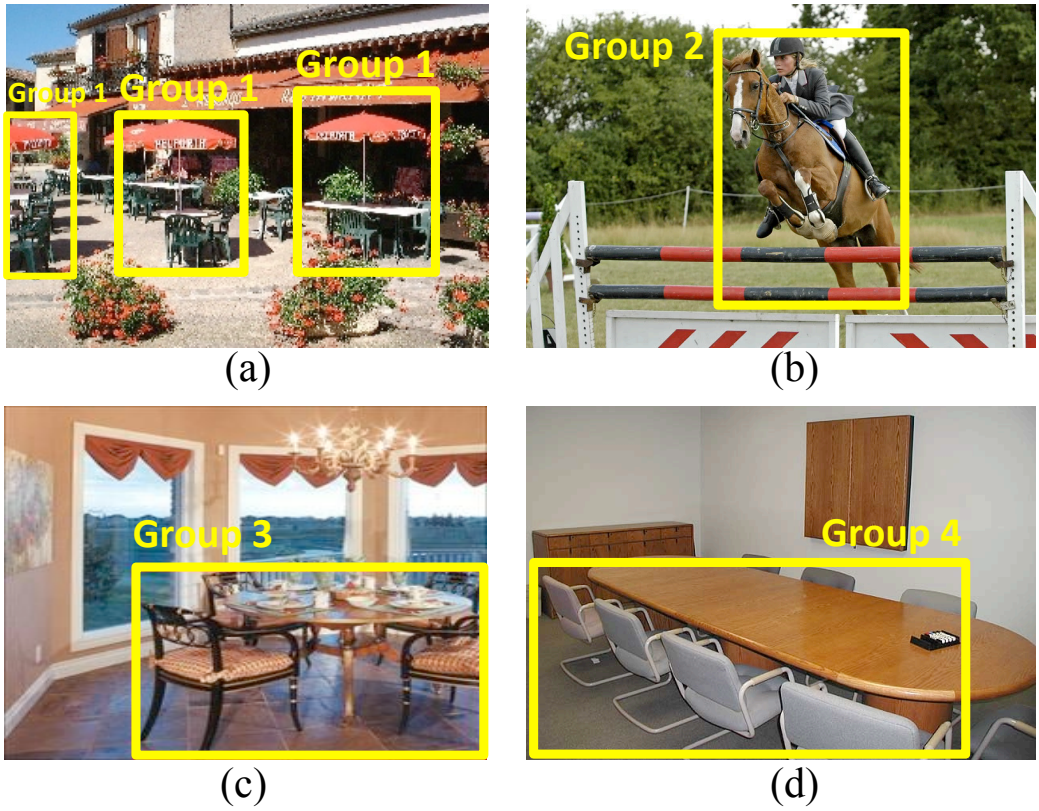


Figure 5.1: We *automatically* discover and model "groups of objects" which are complex composites of objects with consistent spatial, scale, and viewpoint relationship across images. These groups can aid detection of participating objects (e.g. umbrella in (a)) or non-participating objects (e.g. fence in (b)) as well as improve scene recognition (e.g. dining room vs. meeting room in (c) and (d)).

step towards deeper scene understanding.

In this chapter, we promote "groups of objects". They are complex composites of two or more objects which have consistent spatial, scale, and viewpoint relationships with each other across images. Because of this consistency, they are likely to be more detectable than the participating objects in isolation which may demonstrate more intra-class appearance variance across images. Hence, detecting the groups of objects can help improve detection of the participating objects. For instance, the group shown in Figure 5.1(a) boosts the performance of an umbrella detector. Even beyond that, groups of objects are likely to corre-

spond to a specific layout of the scene. They can thus provide strong contextual cues for where *other* objects in the scene are likely to be present. For instance, a group capturing a person on a jumping-horse as seen in Figure 5.1(b) can aid the detection of a fence. Moreover, groups of objects can also better discriminate among scenes that share similar participating objects, but in different configurations such as dining room and meeting room in Figure 5.1(c) and (d).

Groups of objects clearly have potential for aiding various visual recognition tasks. But where do these groups of objects come from? It is not feasible to manually compile a list of all groups of objects with arbitrary numbers of participating objects that we see in the wide variety of scenes in the visual world around us. On the bright side, what the advent of crowd-sourcing services and visual media on the web has given us is large datasets such as PASCAL [28] and SUN [147] that contain many natural images richly annotated with object categories.

In this chapter, we *automatically* and efficiently discover a complete and compact set of object groups containing arbitrary numbers of participating objects. We leverage images annotated with object categories to do so. We build a 4-dimensional transform space modeling spatial location, scale and viewpoint of objects. Objects demonstrating consistent interactions along these dimensions across images are mapped to the same region in this transform space. This space is efficiently mined to discover recurring groups of objects. We model these groups of objects via the deformable part-based model, allowing us to detect these groups in novel images. These detections can now be used as contextual cues for participating or non-participating individual object detection, or for scene categorization.

The contributions in this chapter are as follows: First, we propose modeling a full spectrum of arbitrarily high-order object interactions for deeper scene understanding. These groups contain objects with consistent spatial, scale and viewpoint relationships between each other. Secondly, we propose an algorithm to automatically discover these groups from images annotated only with object labels. We then model the groups using the existing deformable part-based object models. Finally, we demonstrate on a variety of datasets that group detections can improve object detection and scene recognition performance. We also show that our discovered groups are semantically meaningful.

5.2 Related Work

We compare and contrast our work to several existing works that exploit object interactions, model visual composites of scenes, or discover co-occurring visual patterns.

Object interactions: Many works exploit contextual interactions between objects [7, 15, 22, 24, 33, 37, 41, 42, 67, 99, 108, 113, 115, 121, 127, 148] for improved recognition. Most of these works only model pair-wise interactions among objects. Even works that go beyond pair-wise interactions (e.g. Felzenszwalb et al. [33]) typically rely on individual object detections as the source of context. With groups of objects, we can also capture higher-order contextual interactions. Furthermore, we model the visual appearance of the groups of objects as a whole, resulting in a more reliable contextual signal.

Visual composites: Several works have explored entities that fall between individual objects and scenes. In some works [107, 142] including our work in Chap-

ter 4, these entities are discovered from unlabeled regions in images. These entities are hence heavily influenced by the particular choice of appearance models used in the discovery process, and are seldom semantically meaningful. We discover groups of objects by exploiting object-level annotations in images. Our groups tend to be semantically meaningful, and are not dependent on the appearance modeling choices that follow. At the other extreme, some works employ a fully supervised approach to learn visual composites. For instance, Xiao et al. [147] label images with 'subscenes'. Sadeghi et al. [119] label a subset of the PASCAL dataset with 'visual phrases' which are either objects performing an action (i.e. objects in a certain pose such as person running), or a pair of objects interacting with each other (e.g. person riding a horse). They rely on a manual list of 17 visual phrases, and are restricted to groups containing at most two objects from a set of 8 categories. Our work here on the other hand automatically discovers groups containing an arbitrary number of objects. As we show in our experiments, we can discover 71 groups containing upto 6 objects from 107 object categories in the SUN dataset that contains images from a wide variety of scene categories.

Finding co-occurring patterns: Several works have looked at the problem of discovering co-occurring patterns across images: be it for discovering hierarchical spatial patterns of visual words in images [109] or discovering segments of foreground objects of interest [48, 73, 118]. In this chapter, we are interested in finding groupings of objects that consistently co-occur at predictable locations, scales and viewpoints with respect to each other. Zhang and Chen [153] propose an efficient algorithm for calculating kernels capturing similarity between pairs of images using translation invariant arbitrarily higher-order visual codeword arrangements. We employ a similar Hough-transform like mechanism,

but apply to the novel task of discovering groups of objects from images annotated with object categories. We extend their proposed translation based "offset space" to a more complex transform space that also incorporates scale and viewpoint. We also propose a soft voting scheme to be robust to quantization artifacts in this transform space.

5.3 Approach

We first describe the desirable properties of groups of objects, and then present our approach to discover them.

5.3.1 Groups of objects

A group of objects contains two or more objects. For objects to belong to the same group, they must co-occur frequently, and have consistent spatial, scale and pose/viewpoint relationships across images. Each object category may participate in multiple groups, and multiple instances of the same object category may participate in the same group. For instance, a table with four chairs arranged around the table may form one group, while a table with two chairs and an umbrella may form another group. Hence, any naive clustering of categories based on co-occurrence or location/scale/viewpoint consistency would not suffice for our purposes. We propose the following approach to discovering a complete set of groups from images annotated with object bounding boxes.

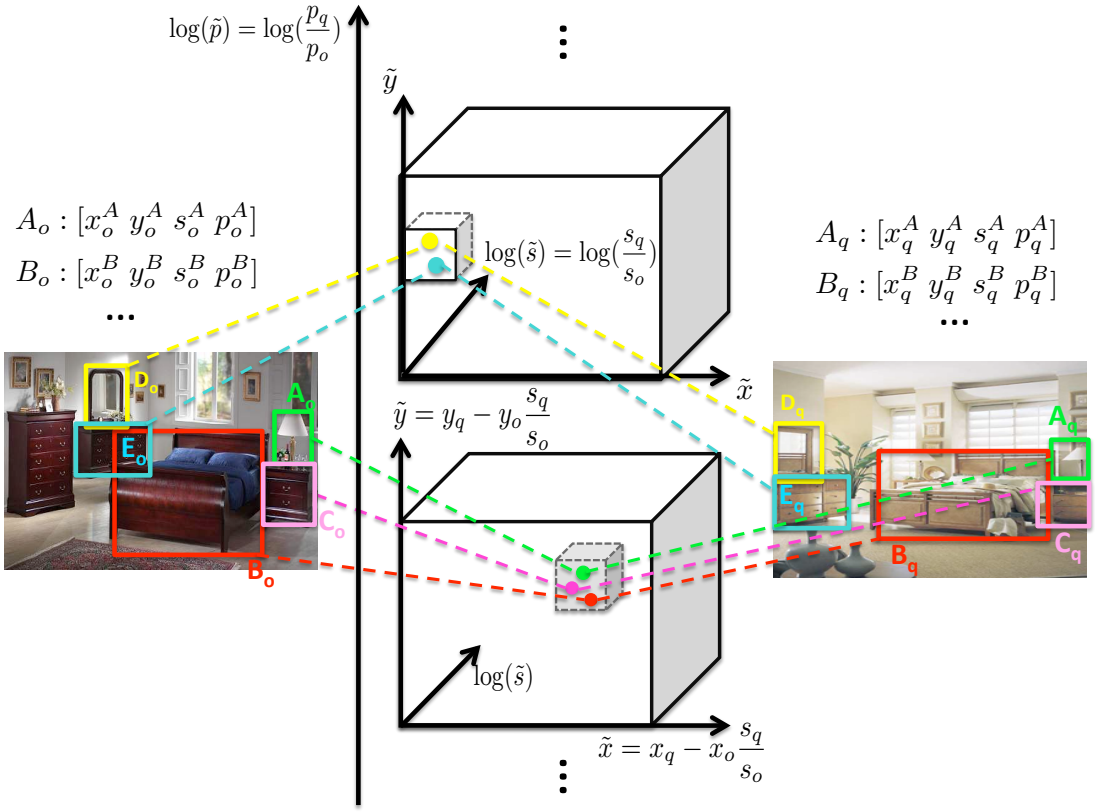


Figure 5.2: Our algorithm for finding high-order recurring patterns of objects utilizes a 4-D transform space. In this example, we note that the three correspondences of objects $(A_o, A_q), (B_o, B_q), (C_o, C_q)$ fall in the same bin in the transform space, thus the objects (A_o, B_o, C_o) and (A_q, B_q, C_q) form a 3^{rd}-order pattern. Similarly, (D_o, E_o) and (D_q, E_q) form a 2^{nd}-order pattern.

5.3.2 Group Discovery Algorithm

Based on our above definition, the task of discovering groups becomes that of discovering consistently occurring object-layout patterns in a set of images. Our intuition is that if two object-layout patterns belong to the same group, they not only contain the same participating object categories, but the objects share similar transformations (in location, scale and viewpoint) that map them from one pattern to the other. For example, in Figure 5.2, it is clear that the object-layout pattern $[A, B, C]$ repeats itself in both images. We know this because the displacement in the location of A between the two images, is the same for B and C. All three translate the same amount between the two images. So if we

look at a transform space that encodes how much an object translates from one image to the other, A, B and C would fall at the same location in the transform space. This forms the intuition behind our approach, except we deal with not only translation, but also scale and viewpoint changes.

Let's first consider a dataset with only two images annotated with object bounding boxes. Let's say the images have a set of objects O and Q respectively. For every object $o \in O$, we are given its category $c(o)$, location $(x(o), y(o))$, scale $s(o)$, and viewpoint $p(o)$, where $(x(o), y(o))$ is computed as the coordinates of the center of the object bounding-box, $s(o)$ is computed as the square-root of the box area, and $p(o)$ is computed as the aspect ratio of the box. Similarly, for any object $q \in Q$, we have $c(q)$, $(x(q), y(q))$, $s(q)$ and $p(q)$.

Now we want to find any co-occurring object-layout patterns between these two images. To do so, we first identify a set of object correspondences $R = \{(o, q) \in O \times Q : c(o) = c(q)\}$. Note that this is a many-to-many mapping: an object in O may correspond to multiple objects in Q, and vice versa. For each correspondence $r \in R$, we construct a transform that describes the location, scale, and viewpoint changes that this correspondence induces: $\mathcal{T}(r) = [\tilde{x}(r), \tilde{y}(r), \tilde{s}(r), \tilde{p}(r)]$. Here $(\tilde{x}(r), \tilde{y}(r))$ denotes the translation of object location, i.e. $\tilde{x}(r) = x(q) - x(o)\frac{s(q)}{s(o)}$ and $\tilde{y}(r) = y(q) - y(o)\frac{s(q)}{s(o)}$, where the factor $\frac{s(q)}{s(o)}$ is used to normalize the translation by the object size[1]. $\tilde{s}(r)$ denotes the scale change, i.e. $\tilde{s}(r) = \frac{s(q)}{s(o)}$ and $\tilde{p}(r)$ denotes the viewpoint change $\tilde{p}(r) = \frac{p(q)}{p(o)}$. This results in a 4-D transform space, as shown in Figure 5.2. To allow for small variance, we quantize the space into discrete bins.

If we have a set of object correspondences $r_1, \cdots, r_n$ where $r_i = (o_i, q_i)$, that

[1]We use the scale of the object as a proxy for estimating the global scale of the scene.

fall in the same bin of the transform space, i.e. share the same transform $\mathcal{T}(r_1) = \cdots = \mathcal{T}(r_n)$, we say that $(o_1, \cdots, o_n)$ and $(q_1, \cdots, q_n)$ form an n^{th}-order object-layout pattern. We represent a pattern via its two instantiations, i.e. $\text{Pa} = \{(o_1, \cdots, o_n), (q_1, \cdots, q_n)\}$.

Note that there may be multiple bins in the transform space that have more than one object in them. For example, in Figure 5.2, we find that, besides the $[A, B, C]$ pattern, $[D, E]$ is also a repeating pattern. Hence between two images, we may have a set of patterns $\text{Pa}_1, \text{Pa}_2, \cdots, \text{Pa}_K$. Naively, one may consider each pattern to be a group of objects. However, note that multiple patterns may correspond to the same group structure (i.e. the participating objects with the same location, scale, viewpoint relationship). For example, assume we find two patterns between the two images: $\text{Pa}_1 = \{(o_1, o_2, o_3), (q_1, q_2, q_3)\}$ and $\text{Pa}_2 = \{(o_1, o_2, o_3), (q_4, q_5, q_6)\}$, as shown in Figure. 5.3(a). The repetition of (o_1, o_2, o_3) in both patterns indicates that (o_1, o_2, o_3), (q_1, q_2, q_3) and (q_4, q_5, q_6) are all instantiations of the same group. Hence we employ a straightforward clustering algorithm to cluster all the discovered patterns to generate a set of groups $\mathcal{G}$. Our algorithm is described in Algorithm 3.

To extend the above approach to a dataset with multiple images, we first find all the patterns between every pair of images. We then cluster the patterns based on the transitivity of patterns across images, as shown in Figure 5.3(b) where Pa_1, Pa_3, Pa_3 should all belong to the same group. We utilize the same Algorithm 1 to find the groups.

Algorithm 3: Generate groups.

$G_1 \leftarrow \text{Pa}_1,\ \mathcal{G} \leftarrow \{G_1\},\ n_G \leftarrow 1$
for $k = 1 : K$ **do**
  $flag \leftarrow 0$
  for $j = 1 : n_G$ **do**
    if $\text{Pa}_k \cap G_j \neq \emptyset$ **then**
      $G_j \leftarrow G_j \cup \text{Pa}_k$
      $flag \leftarrow 1$
      break
    end if
  end for
  if $flag == 0$ **then**
    $n_G \leftarrow n_G + 1$
    $G_{n_G} \leftarrow \text{Pa}_k$
    $\mathcal{G} \leftarrow \mathcal{G} \cup G_{n_G}$
  end if
end for

5.3.3 Soft Voting

There are still two remaining concerns: (1) The above approach as described is sensitive to the quantization of the transform space; (2) Insisting that all participating objects in a group should be present in every instantiation of the group in images is not realistic. Not only does this reduce the instantiations of groups with many objects, it also results in the clustering algorithm discovering many similar and redundant groups. To address these problems, we propose a soft voting scheme for group discovery.

First, to alleviate the effect of hard quantization, instead of each object correspondence falling in only one bin in the transform space (as described above),

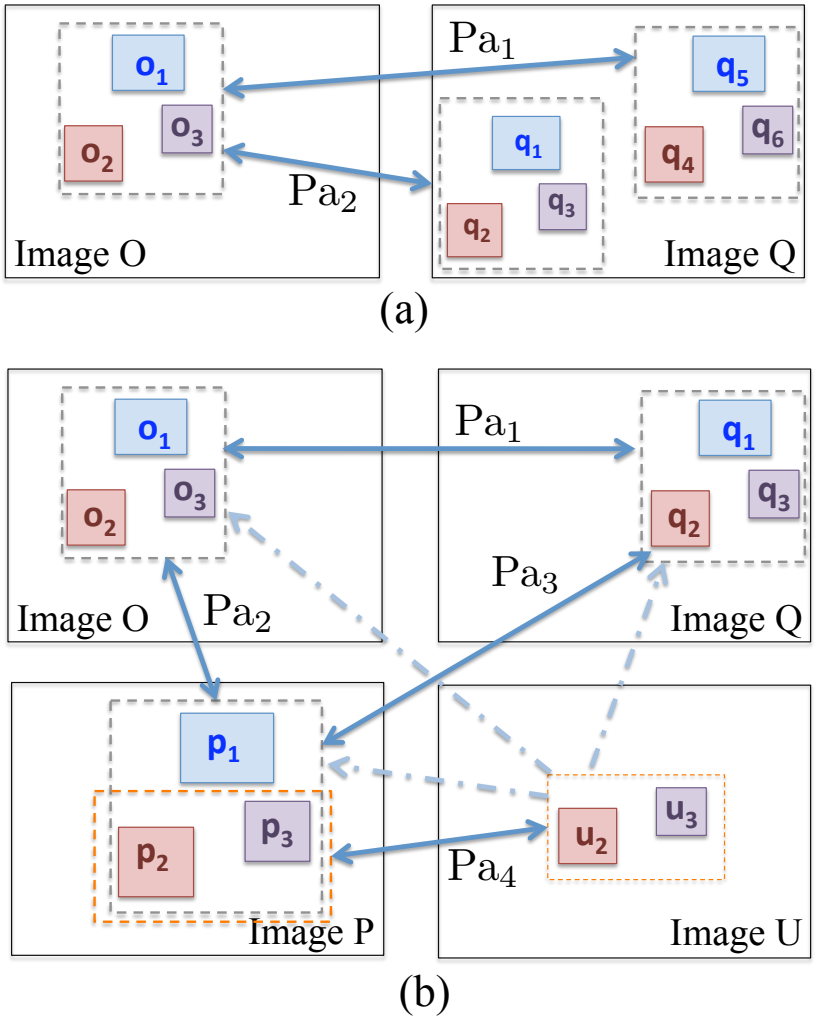


Figure 5.3: Examples of multiple patterns to be combined into a group.

we allow it to vote for neighboring bins weighted by a 4-D gaussian filter with a standard-deviation of 1, indicated by the circles surrounding the object correspondences in Figure 5.4. Note that we show the 2-D transform space $(\tilde{x}, \tilde{y})$ only for ease of illustration. Our implementation uses a 4-D transform space. For each bin, we accumulate the soft votes from all object correspondences, as shown in the heat map in Figure 5.4. We then use non-maximum suppression to find the locations of the peaks. Each object correspondence is assigned to the peak it contributes to the most. A peak that gets n object assignments corresponds to a n^{th}-order pattern. After finding the co-occuring patterns, we apply

the same clustering Algorithm 1 to find groups.

Secondly, to deal with the issue of missing participating objects, we employ a post-processing scheme that allows lower-order group instantiations to be merged with instantiations of corresponding higher-order groups. We do so if only one participating object in the high-order group is missing. For example, in Figure 5.3(b), pattern Pa_4 is an instantiation of a 2^{nd}-order group and patterns $\mathrm{Pa}_1, \mathrm{Pa}_2, \mathrm{Pa}_3$ are instantiations of a 3^{rd}-order group. Since (p_2, p_3) in the 2^{nd}-order group also participates in (p_1, p_2, p_3) in the 3^{rd}-order group, we absorb (u_2, u_3) into instantiations of the 3^{rd}-order group (but with one participating object missing). Note that (u_2, u_3) is no longer considered to be an instantiation of the 2^{nd}-order group.

For a general case, if an n^{th} order group has instantiations where n–1 of the participating objects also participate in an $(n–1)^{th}$ order group, we let the n^{th} order group absorb the instantiations of that $(n–1)^{th}$ order group. We perform this process sequentially on all groups with more than 2 objects, starting with the highest-order groups. We then compute the frequency with which each participating object is present in the group instantiations. We prune objects that participate less than 50% of the time, effectively reducing the order of the group. If the resultant lower-order group already exists, the instantiations are merged. Finally, we only keep groups with more than 30 instantiations in the training data in order to have enough positive samples for training group models as described in Section 5.3.4.

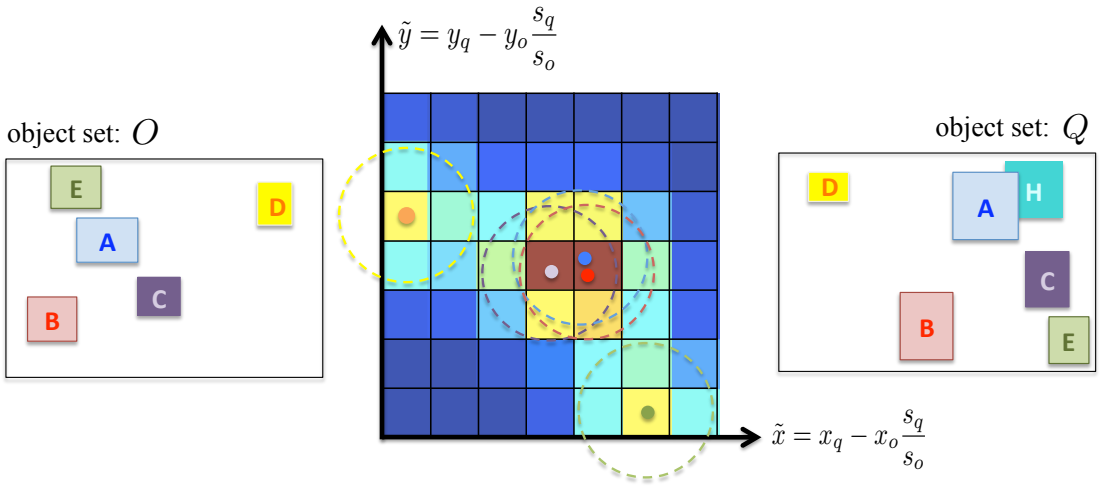


Figure 5.4: An example of object correspondences voting for neighboring bins in the 2-D transform space. The heat map depicts the accumulated soft votes in each bin.

5.3.4 Group Detection

We model the appearance of groups of objects via object models similar to [107, 119] and also the previous work in Chapter 4. This allows us to utilize any off-the-shelf object detector to detect groups. In our experiments we use the deformable part-based model [33]. Specifically, we use the code made available at [32] with default parameter settings to train 4-component group detectors. We now describe how we generate the bounding boxes to train our groups-of-objects detectors.

We generate a bounding box for each instantiation of the groups in the images. If the instantiation has all the participating objects, we generate a bounding box that is the smallest box that encompasses all participating objects. Note that using all instantiations of a group across the dataset, we can estimate the mean location, scale and viewpoint of any participating object with respect to the group. So if any of participating objects are missing we hallucinate the the missing object using these statistics. We then generate a box that encompasses

all objects (including the hallucinated one).

5.3.5 Computational Efficiency

Our approach of discovering groups is quite efficient. The computational cost of finding *all* co-occurring patterns of an *arbitrary* order between a pair of images is linear in the number of object correspondences. Note that these can at most be quadratic in the number of objects in both images if *all* objects within both images are the same category. In practice, the number of object correspondences between two images is small and often less than the number of objects in each image. Our matlab implementation takes less than $3ms$ to find all co-occurring patterns in a pair of densely labeled images on a Macintosh machine with 2.66GHz CPU. Finding all patterns from all pairs of 1434 training images in UIUC phrase dataset *sequentially* took about 50 minutes (obviously, this process is highly parallelizable). Clustering these patterns into groups (including soft-voting) took another 20 minutes. After finding the group instantiations, the computational cost of training group detectors simply depends on the internal learning algorithm of the detection technique being adopted. In the testing stage, the computational cost of detecting the groups simply depends on the internal inference algorithm of the detection technique.

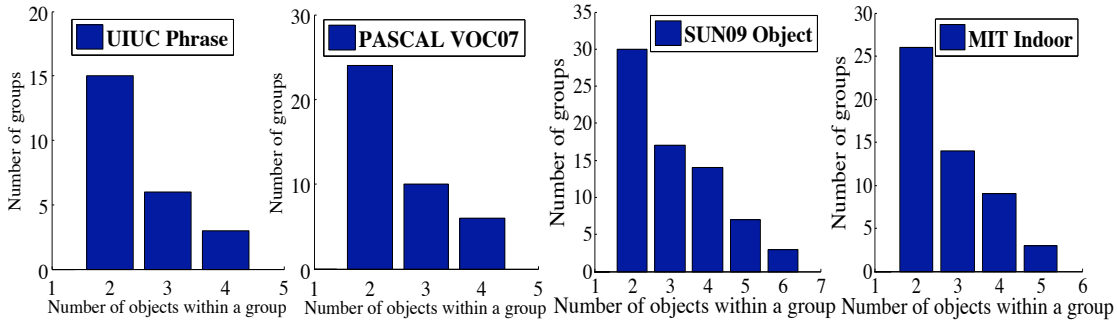


Figure 5.5: Distribution of the number of objects within our automatically discovered groups of objects using four datasets. We can discover a diverse set of high-order groups.

5.4 Experiments and Results

5.4.1 Auto-Discovery of Object Groups

We perform the group discovery on four datasets: UIUC phrase dataset [119], Pascal VOC 2007 dataset [28], SUN09 object dataset [15] and MIT indoor dataset [111]. Examples of object groups discovered by our algorithm for various datasets are given in Figure 5.6. Figure 5.5 also provides the histograms over the object numbers within a group.

UIUC phrase dataset is a subset of the PASCAL dataset. It contains 2769 images labeled with 8 of the 20 PASCAL categories. In addition to the object category annotations, it contains bounding box annotations for a manually generated list of 17 phrases. 12 of these phrases describe interactions between two objects (e.g. person riding horse) and 5 describe a single object performing an action (e.g. dog running). Since the goal of our work is to model groups of more than one objects, we focus on the 12 phrases. Our algorithm discovers 24 groups in this dataset. Our groups contain 2 to 4 objects, as shown in Figure 5.5.

We wish to evaluate how well our automatically discovered groups correspond to the hand-generated list of 12 groups containing two objects. To de-

termine if one of our groups 'matches' one of the phrases in the dataset, we compare the bounding boxes automatically generated by our approach for that group, to the hand-annotated bounding boxes for the phrase. If more than 75% of our bounding boxes have more than 50% intersection-over-union overlap with the hand annotated bounding boxes, we assign our group to that phrase. Note that each group can match only one phrase, but multiple groups can match the same phrase. We find that every phrase has at least one matched group. However, if we use a lower dimensional transform space (e.g. $(\tilde{x}, \tilde{y})$ or $(\tilde{x}, \tilde{y}, \tilde{s})$), different phrases (e.g. 'person riding horse' and 'horse and rider jumping') would be grouped together. Apart from phrases, our groups also capture other concepts such as two-people on a sofa. We merge the bounding boxes from all groups that match the same phrase. We now have a one-to-one correspondence between the phrases and the matched groups. In Table 5.1 we report the percentage of the phrase bounding boxes covered by our groups. We find a large proportion of the hand annotated bounding boxes have been discovered by our automatic approach. We train detectors for detecting the manually labeled phrases, but using our automatically discovered group bounding boxes. As seen in Table 5.1, the performance is comparable to and sometimes even superior to training a detector using the manually labeled bounding boxes! We use the same test settings as in [119]: roughly 50 positive and 150 negative images. This confirms that our approach can find semantically meaningful groups in an automatic manner.

PASCAL VOC 2007 dataset contains 9963 images with annotations for 20 object categories. We discover 40 groups containing 2 to 4 objects (Figure 5.5). We use these groups as contextual cues for improving object detection performance (Section 5.4.2). **SUN09 object dataset** contains 12059 images annotated

Phrase Names	Ratio covered by groups	AP (trained by manual labels) [119]	AP (trained by discovered groups)
Person next to bicycle	81.7%	46.6	43.5
Person lying on sofa	72.9%	24.9	25.2
Horse and rider jumping	80.0%	87.0	86.5
Person drinking from bottle	91.7%	27.9	30.3
Person sitting on sofa	69.1%	26.2	24.8
Person riding horse	77.7%	78.7	77.3
Person riding bicycle	82.3%	66.9	66.1
Person next to car	64.2%	44.3	41.2
Dog lying on sofa	85.1%	23.5	25.5
Bicycle next to car	84.0%	44.8	49.6
Person sitting on chair	95.2%	20.1	21.5
Person next to horse	68.2%	35.1	34.5
MEAN	79.3%	43.8	43.8

Table 5.1: Column 1: the ratio of training examples for a phrase covered by our corresponding groups. Column 2: detection performance of detectors trained using manually labeled phrase bounding boxes in [119]. Column 3: detection performance of detectors trained using our automatically discovered group bounding boxes. Our automatically discovered groups match the manually annotated phrases very well. (APs measured in %.)

with 107 object categories, the largest dataset of its kind. Our algorithm discovers 71 groups containing 2 to 6 objects (Figure 5.5). Again, we use these groups as contextual cues for improving object detection (Section 5.4.2). **MIT indoor dataset** contains 15613 images from 67 scene categories and 423 labeled object categories. Since only 2743 images in the dataset have object annotations, we select 15 categories that have more than 50 training images annotated, as listed in Table 5.5-Row 1. We utilize 152 object categories present more than 20 times in images of the 15 scene categories. We discover 52 groups containing 2 to 6 objects. We use these groups to improve scene recognition performance as described in Section 5.4.3.

5.4.2 Object Detection

We use the deformable part-based model [33] to train detectors for all the individual objects of interest (OOI) and the groups. We use the contextual re-scoring

Figure 5.6: Examples of our automatically discovered groups of objects from four datasets. Instantiations of the same group depict the same objects with consistent spatial, scale, and viewpoint relationships. They often have the same semantic meaning. At the 4^{th} column of the 4^{th} row, we also show a failure case where the instantiations do not have the same semantic meaning. This is because objects interact with each other in complex ways, which may not always be captured by our 4-dimensional transform.

scheme used by Felzenszwalb et al. [33]. We re-score a candidate OOI detection using a classifier that incorporates the highest detections of groups of objects in the image. We evaluate the resultant improvements in object detection performance on three datasets: UIUC phrase dataset, PASCAL VOC 2007 dataset, and SUN09 object dataset.

UIUC phrase dataset. Table 5.2 compares the object detection improvement by using our automatically discovered object groups as contextual cues, as opposed to using detectors for the manually defined phrases [119] as context. We also compare with using other individual object categories as contextual information as in [33]. The same contextual re-scoring scheme is used for all approaches.

We see that using our automatically discovered groups outperforms the

	bike	bottle	car	chair	dog	horse	pers	sofa	MEAN
Base w/o context [33]	57.0	7.0	25.8	11.1	5.6	49.3	25.7	14.1	24.5
Object context ([33])	58.8	9.3	**33.1**	13.4	5.0	53.7	27.9	19.8	27.6
Phrase context ([119])	60.0	9.3	32.6	**13.6**	**8.0**	53.5	28.8	22.5	28.5
Group context	**63.5**	**10.7**	32.5	13.2	**8.0**	**54.6**	**30.6**	**24.9**	**29.8**

Table 5.2: Average precision (AP) for all 8 categories in UIUC phrase dataset, mean AP across all categories. Methods: Baseline without context; Object context (rescoring using other objects); Phrase context (rescoring using the manually defined phrases); Group context (rescoring using our automatically discovered object groups).

other two methods in 5 out of 8 categories and performs comparably for the remaining 3 categories. There are two main reasons for our approach having better performance: (1) Unlike the phrases [119], our groups contain more than just 2 objects. For instance, we have a group composed of two horses and two persons, and another group containing four persons. (2) Our groups explicitly model the spatial and scale relationship between objects, thus resulting in more robust appearance models themselves. For instance, in Figure 5.6, we have two groups both containing a person and a bottle, but with different spatial interactions. These are lumped together in [119] as "person drinking bottle" resulting in large intra-class variance in appearance.

PASCAL VOC 2007 dataset. Table 5.3 shows the results of using different types of contextual information to improve the detection performance of objects. We compare our method with the baseline method without using context, the method of using other objects as context [33] and the state-of-the-art method of using contextual-meta-objects (CMO) recently proposed in Chapter 4 as context. CMOs are contextually relevant regions for each object category that are automatically discovered by using that object as an anchor point and exploiting surrounding unlabeled regions. Our groups on the other hand have access to more annotations but are unable to leverage unlabeled regions that may be consistent. Hence CMOs and our groups can be viewed as being complementary

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbik	pers	plant	sheep	sofa	train	tv	MEAN
Base w/o context [33]	28.9	59.5	10.0	15.2	25.5	49.6	57.9	19.3	22.4	25.2	23.3	11.1	56.8	48.7	41.9	12.2	17.8	33.6	45.1	41.6	32.3
20OOI ([33])	**31.2**	61.5	**11.9**	**17.4**	27.0	49.1	59.6	23.1	23.0	26.3	24.9	12.9	60.1	**51.0**	43.2	**13.4**	18.8	36.2	**49.1**	43.0	34.1
CMO	30.5	60.1	11.2	17.0	26.7	**49.7**	59.1	23.3	23.4	26.9	29.3	**13.2**	59.7	49.3	43.0	**13.4**	20.4	37.8	46.8	**43.3**	34.2
Group	29.5	**62.4**	10.8	16.4	**28.3**	**49.7**	**60.7**	**23.8**	**24.5**	**27.2**	**31.3**	**13.2**	**61.0**	49.2	**43.5**	12.7	**20.9**	**38.8**	45.3	42.6	**34.6**
Groups+OOIs+CMOs	**31.5**	**63.0**	**12.6**	**18.1**	**29.0**	**51.7**	**61.4**	**25.0**	**24.9**	**28.0**	**31.4**	**14.1**	**61.5**	**51.4**	**44.0**	**14.6**	**21.2**	**39.4**	**49.1**	**44.3**	**35.8**

Table 5.3: AP (%) for 20 categories in PASCAL VOC 2007 and the mean AP across 20 categories. Our proposed groups of objects outperform and are complementary to existing sources of context for object detection. Best single-context performance and best overall performance are in bold.

sources of context. Overall, using the discovered groups as context achieves better average performance across the 20 object categories over the other two methods. Furthermore, combining these various contextual cues further boosts performance.

SUN09 object dataset. SUN09 is a very challenging recent dataset containing complex scenes with many object categories and large within-class variance. Table 5.4 gives the mean average precision across the 107 object categories. We compare five different methods: (1) Base w/o context: the individual object detector trained using the deformable part-based model [33] trained on the same additional dataset as the state-of-the-art [15]. (2) H-context: the tree based hierarchical contextual model proposed in [15] which models the inter-object contextual interactions as well as the global scene context. (3) OOIs: using all object categories to provide contextual information through re-scoring [33]. (4) Groups: using the detected groups to provide contextual information for object detection using the same re-scoring. (5) OOIs+Groups: using both objects and groups for re-scoring. Table 5.4 shows that even with such a simple re-scoring algorithm, using the groups as context outperforms the state-of-the-art algorithm. Although the state-of-the-art algorithm also models the spatial and scale relationship between objects, it relies on the performance of the individual ob-

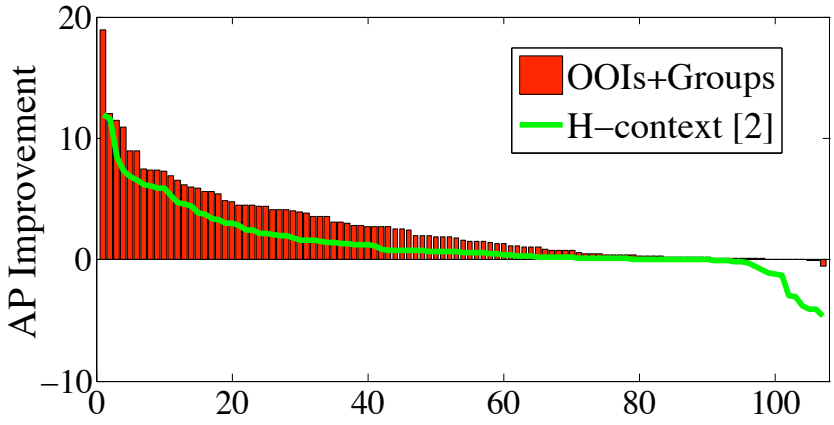


Figure 5.7: Improvement of different context methods over the baseline detectors on SUN09 object dataset. Object categories are sorted by the improvement in average precision (AP). (AP measured in %)

ject detectors. Some contextually relevant categories such as flowers and vases are both difficult to detect in isolation and can not benefit each other. However, the appearance of the flowers-vase group is more consistent and can be reliably detected. We find that our simple re-scoring method using objects and groups as context outperforms the state-of-the-art algorithm on 88 among the 107 categories on this challenging dataset, and performs comparably on the remaining. Figure 5.7 shows the improvement in average precision for each object category sorted by the improvement over the baseline. We note that our method rarely hurts the baseline (i.e. falls below zero) while the state-of-the-art does so to several categories. Our method also achieves larger improvement in a large number of categories.

Figure 5.8 shows that by increasing the highest order of groups to be used for providing contextual information, the mean performance over all object categories increases. This confirms the usefulness of high-order groups and hence the need for an automatic approach to discover these groups.

	mean AP
Base w/o context [33]	7.06
H-context [15]	8.37
OOIs	8.34
Groups	9.06
OOIs+Groups	9.75

Table 5.4: The mean average precision (AP) across 107 object categories in SUN09 object dataset using different methods. (APs measured in %)

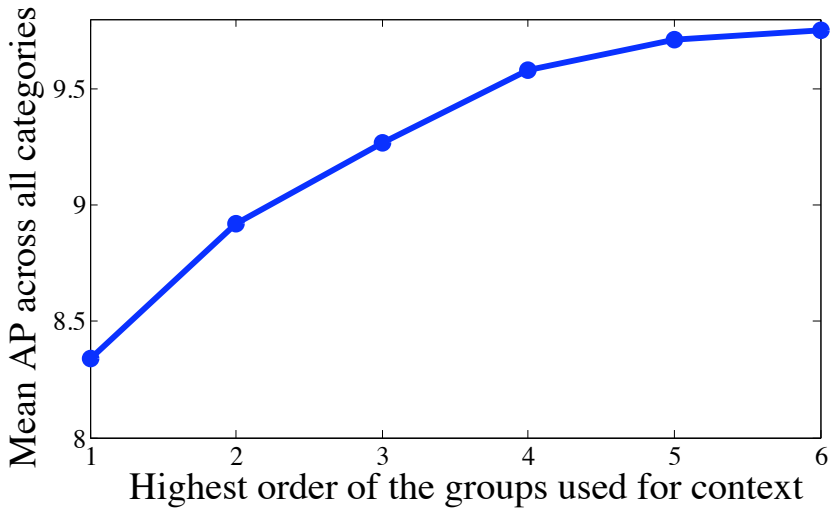


Figure 5.8: The mean APs (%) across the 107 object categories in SUN09 dataset as higher-order groups are included. Order = 1 indicates using the individual objects as context. Clearly, higher order groups provide useful contextual information for object detection.

5.4.3 Scene Categorization

Intuitively the object groups stand as more structured components in a scene than individual objects. In this section, we make use of the object groups to improve the scene categorization performance. We consider 15 of the 67 categories in MIT indoor scene dataset [111] as described earlier.

To analyze the usefulness of object groups to represent a scene as opposed to just individual objects, we conduct an experiment of scene classification based on the groundtruth annotated objects in the images (automatic experiments fol-

	GIST-color	SP	DPM	Objects (OBJ)	Groups (GRP)	GIST-SP-DPM-OBJ	GIST-SP-DPM-OBJ-GRP
airportIn	10.0	30.0	25.0	**40.0**	35.0	**40.0**	**40.0**
artstudio	10.0	10.0	40.0	45.0	**50.0**	**55.0**	**55.0**
bakery	26.3	**57.9**	47.4	47.4	47.4	63.2	**68.4**
bar	16.7	**50.0**	33.3	38.9	33.3	**44.4**	38.9
bath_rm	72.2	55.6	**83.3**	61.1	77.8	83.3	**88.9**
bed_rm	47.6	71.4	9.5	61.9	**76.2**	33.3	**61.9**
bookstore	15.0	50.0	**65.0**	55.0	60.0	60.0	**70.0**
class_rm	10.0	30.0	25.0	**40.0**	35.0	**40.0**	**40.0**
corridor	**83.3**	55.6	**83.3**	50.0	**83.3**	83.3	**88.9**
dine_rm	66.7	**76.2**	**76.2**	66.7	66.7	**81.0**	**81.0**
kitchen	22.2	11.1	27.8	27.8	**33.3**	50.0	**61.1**
living_rm	5.0	0.0	25.0	25.0	**30.0**	20.0	**25.0**
mtg_rm	22.7	27.3	**81.8**	27.3	63.6	77.3	**81.8**
office	14.3	0.0	38.1	19.1	**33.3**	33.3	**42.9**
warehouse	33.3	**61.9**	47.6	42.9	47.6	47.6	**52.4**
MEAN	33.8	40.0	49.7	43.4	**53.6**	55.9	**61.8**

Table 5.5: Classification rates (%) for the 15 scene categories in MIT Indoor dataset and the mean classification rate (%) across 15 categories. Best single-approach performance and best combined performance are in bold. Our proposed groups of objects boost the performance of scene recognition.

low next). We compare three image descriptors which are classified by an RBF-kernal SVM. The first is a 152-D vector indicating the occurrence of each object, the second is an analogous 52-D vector for our groups, and the third is a concatenation of both. The average accuracy for the 15 class scene categorization is respectively 81.5%(object), 84.5%(group), and 89.0%(object+group). This demonstrates the benefit of using groups of objects, as well as the complementary nature of objects as groups.

In the following experiments, we use the same training / testing split as in [111], where each scene category has 80 training images and 20 testing images. We compare different approaches for scene categorization: (1) **GIST-color** [106]: features are computed by concatenating the three 320-dimensitional GIST descriptor of the RGB channels of the image, followed by the one-vs-all SVM classifiers with RBF kernel. (2) **Spatial Pyramids (SP)** [71]: We compute the spatial pyramid features with the implementation provided by [71]. We use a vocab-

ulary of size 200 and three levels in the pyramid, followed by one-vs-all SVM classifiers with the histogram intersection kernel. (3) **Deformable Part-based Model (DPM)**: Pandey et al. [107] recently proposed the use of a deformable part-based model for a scene categorization, which implicitly captures concurrent regions within a scene. (4) **Objects (OBJ)**: we represent an image with the detected individual objects. We note that due to the partial object labeling of the training images and the large variance of the object appearance, it is difficult to train a robust object detector with the limited number of positive samples. To boost this baseline, we use 152 object detectors trained with an additional annotated dataset in [15] to detect objects. We apply the object detectors on all images. For each image, we form a 152-dimensional feature vector with each dimension indicating the highest score among the detections of each object category on the image. (We also tried using the histogram of detected objects as input, but achieved lower performance.) RBF-kernel SVM classifiers are utilized for classification. (5) **Groups (GRP)**: We train detectors for our groups with the very limited positive training samples. We apply the group detectors on all images. Each image is represented as a 52-dimensional feature vector with each dimension indicating the highest score among the detections of each group on the image. Again, one-vs-all SVM classifiers with RBF kernel are utilized for classification. (6) **GIST+SP+DPM+OBJ**: We combine all the above methods except the groups by multiplying the softmax-transformed outputs of the SVMs from each method similar to [107].(7) **GIST+SP+DPM+OBJ+GRP**: We combine all the above methods including the groups.

Results. We summarize the results for the different methods in Table 5.5. We see that our proposed groups of objects outperform all methods. Many scenes (e.g. meeting room and dining room) may contain similar objects and can be confus-

ing. On the other hand, global appearance of scenes may vary (e.g. the locations of cabinets or table-chair sets in dining rooms). Groups of objects (e.g. configuration of tables and chairs) seem to hit the right balance between the generalization and discriminative power. We also note that combining the object group results with those from the other approaches further improves performance.

5.5 Summary

In this chapter, we propose to model group of objects, which are high-order composites of objects with consistent spatial, scale, and viewpoint relationships with respect to each other across images. Manually listing all possible groups of objects is not feasible for groups containing arbitrary number of diverse object categories in a wide variety of scenes. We propose a novel Hough-transform based approach to efficiently discover the groups of objects from images annotated with object categories. We model groups of objects via deformable part-based models. Our extensive experiments on 4 challenging datasets show that the detection of groups can improve both object detection and scene categorization, and outperform multiple state-of-the-art methods. This work has first appeared in the following publication: Li, Parikh, Chen [88].

In the current group discovery algorithm, we only allows variations in translation, scale, and approximated viewpoint. In the future, we would like to consider more variations, such as rotation and occlusion. Furthermore, besides using the detected groups as contextual cues for object and scene recognition, we also like to incorporate the detected groups as well as objects to describe images.

CHAPTER 6

APPLICATIONS: IMAGE AESTHETICS, VIDEO ANALYSIS, AND ROBOTICS

In this chapter, we present applying the proposed FECCM models onto various applications, in additional to the natural scene understanding scenario that has already been evaluated in Chapter 2. We consider three scenarios: images aesthetics, video analysis, and robotic system. Some initial efforts of these works have first appeared in the following publications: Li, Gallagher, Loui, Chen [81], Li, Chen, Dunker, Cremer, Chen [78], Li, Lin, Yu, Chen [85], Li, Wong, Xu, Saxena [89], and Li, Kowdle, Saxena, Chen [83, 84].

6.1 Image Aesthetics

Visual aesthetic quality is a measure of visually perceived beauty. Judgement of the visual aesthetic quality of images is highly subjective. However, some images are often believed, by consensus, to be visually more appealing than others. This serves as one of the principles in the emerging research area of computational aesthetics. Our goal here is to explore computational solutions to automatically infer the aesthetic quality of images. The greatest challenge in this research lies in the gap between low-level image properties and the high-level human perception of aesthetics.

In recent years, there has been rapidly increasing interest in how to computationally predict the aesthetic quality of images. Most existing works [14, 19, 60, 61, 79, 81, 133, 149] address the challenge of automatically predicting

image aesthetic qualityusing their visual content as a machine learning problem. Generally, these techniques first extract computational features from the images and then learn the mappings from the features to the human-rated aesthetic quality through learning techniques. Previous research focus on designing features to represent some high-level concepts which are highly correlated to the image aesthetics. The intuition of feature extraction often comes from concepts in psychology, photography, or human studies. For example, Ke et al. [61] and Datta et al. [19] extract low-level features to represent the high-level properties like color preferences, the image's composition, object shape, and so on. Addressing digital images of paintings, Li and Chen propose a set of features motivated from psychology, art, and controlled human study, to predict common people's aesthetics judgment towards different impressionistic paintings [79]. Recently Dhar et al. [23] propose adding the high-level describable image attributes to represent the image appeal. The authors utilize existing algorithms to extract high-level attributes such as the presence of salient objects, the presence of faces, and the scene category of the image. Their framework is equivalent to the CCM structure with simply the aesthetics assessment task on the second layer. The improvement in performance demonstrates the power of utilizing high-level semantic attributes and encourages closer collaboration between the aesthetic quality assessment task with other tasks that are dedicated to the holistic scene understanding. The more we understand the content and structure of the image, the better performance we can achieve in the quality prediction.

In our work, we apply the FE-CCM algorithm to better optimize the performance of the aesthetic quality assessment task. We consider the aesthetic quality assessment as a two-class classification problem: classifying an image as high

aesthetic quality or low aesthetic quality. In the model, we compose the following tasks: scene categorization, object detection, depth estimation, saliency detection, and aesthetics prediction, as shown in Figure 6.1. The implementation of scene categorization, object detection, depth estimation and saliency detection is the same as Chapter 2, except for object detection, we use the PASCAL VOC 2007 dataset [28] and consider all 20 object categories. The second-layer aesthetic quality assessment classifier is a logistic classifier, taking all the output of the first-layer classifiers and the original features for aesthetics prediction as input. The first-layer aesthetic classifier is a linear SVM classifier. The features are described in the following.

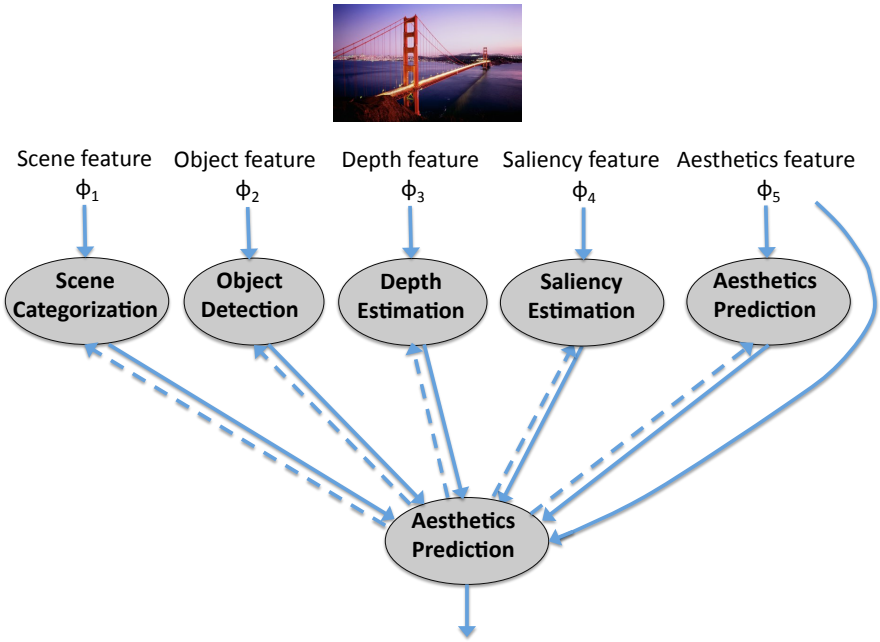


Figure 6.1: The FE-CCM framework of composing multiple tasks for aesthetic quality assessment.

Color statistics and distribution: The image is represented in the HSV color space. We compute the average Hue, Saturation, and Value across all pixels in the image. Besides, we also compute the 3-dimensional color histogram, by quantizing the three color channels into $16 \times 8 \times 8$ bins.

Hue count: The hue count of an image is a measure of the image's simplicity [61]. A K-bin histogram is computed on the hue values of the pixels. Note only pixels with brightness values in the range $[0.15, 0.95]$ and saturation larger than 0.2 are considered. This is because a pixel not satisfying these requirements looks close to gray color no matter what hue value it has. The hue count is the number of bins with values larger than a certain threshold t, i.e.,

$$N_{hue} = \|\{i|H_{hue}(i) > t, 1 \leq i \leq K\}\|, \tag{6.1}$$

where $\|\cdot\|$ indicates the number of elements in a set. We set $K = 20$ and t is 20% of the maximum number in all bins.

Hue harmoniousness: We utilize the models from Matsuda's Color Coordination [100] to evaluate the extent of color harmony in an image, as shown in Figure 6.2. In Figure 6.2, the type-N model corresponds to gray-scale images while

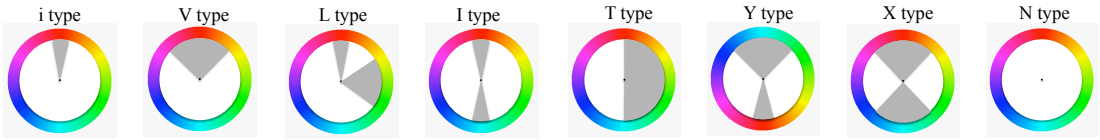


Figure 6.2: The hue harmony models.

the other seven models, each of which consists of one or two sectors, are related with color images. Gray regions indicate that the respective model prefers the pixel hues to agglomerate within the specific regions. Since all the models can be rotated by an arbitrary angle, what they are measuring is the relative relationship of the hues in the image rather than the specific color distribution modeled by the hue histogram itself. To evaluate the color harmony, given an image, we fit each of these models to the hue histogram of the image. We first define $T_k(\alpha)$ as the k^{th} hue model rotated by an angle α and $E_{T_k(\alpha)}(p)$ as the hue in the model

$T_k(\alpha)$ that is closest to the hue of the p^{th} pixel $h(p)$, that is,

$$E_{T_k(\alpha)}(p) = \begin{cases} h(p) & \text{if } h(p) \in G_k \\ B_{nearest} & \text{if } h(p) \notin G_k \end{cases}, \tag{6.2}$$

where G_k is the gray region of model $T_k(\alpha)$ and $B_{nearest}$ is the hue of the sector border in model $T_k(\alpha)$ that is closest to the hue of pixel p. Further we define the distance between the hue histogram and the k^{th} model rotated by α as the function below.

$$F_{k,\alpha} = \sum_p \|h(p) - E_{T_k(\alpha)}(p)\| \cdot s(p) \tag{6.3}$$

where $\|\cdot\|$ refers to the arc-length distance on the hue wheel and $s(p)$ is the saturation of pixel p, which appears here as a weight since the difference between colors with low saturation is perceptually less noticeable. Now in order to fit the k^{th} model with the current image, we look for the best rotation angle α_k^* that minimizes the function $F_{k,\alpha}$, that is,

$$\alpha_k^* = \underset{\alpha}{\operatorname{argmin}}(F_{k,\alpha}). \tag{6.4}$$

We repeat the above operations for all harmony models and find out how each of them fits with the image, which forms a set of harmony scores $\{F_{k,\alpha_k^*}\}, k = 1, \ldots, K$ and serves as the color harmoniousness feature.

Lightness contrast: We measure the lightness contrast as follows. Let H_v be the histogram of the lightness values of the pixels. Based on the histogram, we first search for the minimal region $[b_l, b_r]$ that centralizes 98% energy of the lightness histogram. Next we define the lightness contrast to be measured as $|b_r - b_l|$.

Golden-section rule: The golden-section rule specifies that the focus (main object of the interest) should be located at one of the four intersections. In photography, this rule is often approximated by the Rule of Thirds, which approximates the golden ratio as $\frac{1}{3}$. Here we measure how well the composition of an

image fits with this rule. We first segment an image into 10 regions using the Graph-cut method based on the RGB color and the pixel location, and also apply saliency estimation to compute the saliency of each pixel. Then we compute the saliency of each region by averaging the saliency scores of its pixels. We then compute the golden-rule based feature as the minimum among the distances between the most salient region and all four golden-section line intersections.

$$f_{\text{golden}} = \min_{i=1,2,3,4} \sqrt{((O_x - P_x^i)^2 + (O_y - P_y^i)^2)} \tag{6.5}$$

where (O_x, O_y) is the coordinates for the center of the most salient region and $\{(P_x^i, P_y^i)|i = 1, 2, 3, 4\}$ is the set of coordinates for the four intersection points. All the coordinates have been normalized by the width and height of the image respectively.

Visual balance: Balance is an important concept related to visual harmony and aesthetics. we first segment an image into 10 regions and consider the top 3 largest regions. We first compute the center (C_x^j, C_y^j) and the respective mass M^j for the j^{th} region $(j = 0, 1, 2)$. We then compute the offset between the weighted average center for these J regions and the image center (C_{im}^x, C_{im}^y), as described below. All the coordinates are normalized respectively by the width or height of the image.

$$f_{bal}^x = \frac{\sum_j M^j C_x^j}{\sum_j M^j} - C_{im}^x \tag{6.6}$$

$$f_{bal}^y = \frac{\sum_j M^j C_y^j}{\sum_j M^j} - C_{im}^y \tag{6.7}$$

Shapes: We first segment an image into 10 regions using the Graph-cut method based on the RGB color and the pixel location, and then compute the following shape features for the largest segment: center of mass (first-order moment),

variance (second-order centered moment), and skewness (third-order centered moment).

Sharpness-blurring distribution: We adopt the method proposed in [61]. We first apply Laplacian filter to the grayscale version of the image, and take the absolute values of the results to generate an edge map $E(x, y)$ for the image. Based on the edge map we want to find out the smallest bounding box that encloses a high ratio (80%) of the edge energy.

Dataset. For aesthetic quality assessment, we use the dataset collected by Datta et al. [19] from Photo.net. The dataset contains 3581 images with aesthetic ratings between 1 and 7 given by the users on Photo.net. We use the average score over the users as the aesthetic score for each image. To divide the images into two categories: high-quality class and low-quality class, we set images with scores higher than or equal to 5.8 into the high-quality class and images with scores lower than or equal to 4.2 into the low-quality class. Our results are reported on 5-fold cross validation.

Results. Figure 6.4 compares the performance of several algorithms: the method using manually-designed features by Datta et al. [19], the method using generic features that are commonly used for other scene understanding tasks such as SIFT and color features by Marchesotti et al. [97], the CCM method proposed by Heitz et al. [53] using our features (conceptually similar to Dhar et al. [23]), and our FE-CCM algorithm. The performance is measured in ROC curve. We note that our FE-CCM algorithm outperforms the other three methods. Especially comparing with the CCM method (red curve), the gain comes from the feedback mechanism in our algorithm. Besides, we also check the weights of the second-layer logistic classifier, and note that the top five highest positive

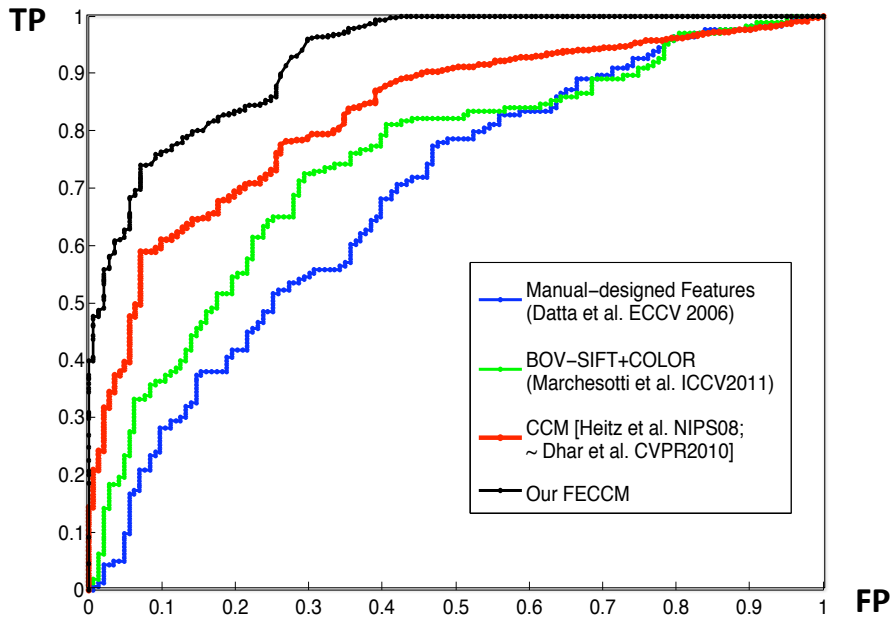


Figure 6.3: The ROC performance for multiple methods on aesthetic quality assessment.

weights are given to the following first-layer outputs: the confidence score of being an ocean scene, the depth of the top-middle region, the fitting score of the harmony model-X, the saliency in the center region, and the lightness contrast. These attributes are also commonly mentioned in previous aesthetics-related works, as they are consistent with human's understanding about important factors that affect the aesthetic quality of an image. Figure 6.3 also gives the images with top-rank and bottom-rank aesthetic quality predicted by our algorithm.

Discussion. In this framework, when we retrain the first-layer classifiers for the tasks of scene categorization, depth estimation, object detection and saliency estimation, we not only utilize the aesthetic data with the predicted values for these first-layer hidden outputs, but also utilize the original dataset having ground-truth for the specific tasks. Therefore, the first-layer classifiers would not learn to be too much away from the original semantic concept. In order to optimize the second-layer aesthetics prediction, the first-layer classifiers

(a) Images with top-rank aesthetic scores

(b) Images with bottom-rank aesthetic scores

Figure 6.4: The examples with the highest aesthetic scores and with lowest scores given by our FE-CCM algorithm. The red frame indicate the wrong classification of the image's aesthetics.

seem to learn biased towards the aesthetics data. If we evaluate the first-layer performance on their original specific dataset, the performance generally drops a little with the feedback mechanism. One interesting thing we note in the person detector is that the overall performance of first-layer person detection drops, however, for the parts of data that the presence of a person only contains the face region, the detection performance increases. This indicates that, with the feedback from the second-layer aesthetics classifier, the first-layer person detector has been driven towards a face detector instead of a more general detector that fires no matter it is a face, a part of body, or the whole body. Furthermore, we observe that the aesthetic score of a portrait image is usually high. This suggests that the feedback is transmitting this correlation between the aesthetic score and the presence of a portrait face. Further investigation into the first-layer "drifts"

would be an interesting direction for future work.

6.2 Video Analysis

6.2.1 Video Mood Classification

In this application, our goal is to classify the mood of a video with only visual features. The types of moods we consider include: Joyous, Sad, Dark, Tense, Sensual, Peace, etc. The ground-truth mood labels for the video clips we study are from experts in the movie analysis area. A naive solution to solve this problem is to extract a group of low level visual features followed by a multi-class classier. However, in our intuition, the mood of a video clip is highly related to many other tasks in video analysis, such as scene classication, human detection, ob ject detection, etc. In this section, we apply the FE-CCM model to compose multiple tasks to help improve the performance of video mood categorization. In the model, we compose the following tasks: scene categorization, face expression classification, video mood categorization, as shown in Figure 6.5. In the following, we describe our implementation for each of our classifiers.

Figure 6.5: Left: overview of the FE-CCM framework on composing three tasks for video mood categorization. Middle and Right: two examples of results from our algorithm.

Scene Categorization. For scene categorization, our goal is to classify an image into one of the 3 categories: Natural, Man-made, and Indoor. We define the output of a scene classifier to be a 3-dimensional vector with each element representing the log-odds score for each category. For the first-layer scene classifier, we use a RBF-Kernel SVM classifier, as used in [135], to classify an image into one of the scene categories. We use a 512-dimensional GIST features [105] as input for the first-layer classifier.

Face Expression Classification. For face expression classification, our goal is to classify the expression of a face into one of the 7 categories: Happy, Sad, Angry, Fearful, Surprised, Disgusted, and Neutral. We define the output of a face expression classifier to be a 7-dimensional vector with each element representing the log-odds score for each category. For the first-layer scene classifier, we use a RBF-Kernel SVM classifier. To extract feature inputs for the first-layer classifier, we first apply Gabor filtering on the normalized facial region, sample the filtering results by applying a mean filtering on feature points inside the face (the feature points are extracted by the ASM method [17]), and then reduce the dimension to 200-dimension using the PCA method.

Video Mood Categorization. For video mood categorization, our goal is to classify an video frame into one of the 9 categories: Peaceful, Sad, Extravagant, Easygoing, Gentle, Thoughtful, Dark, Coarse, and Powerful. We define the output of a mood classifier to be a 9-dimensional vector with each element representing the log-odds score for each category. For the first-layer mood classifier, we use a RBF-Kernel SVM classifier, to classify a frame into one of the mood categories. We use a 230-dimensional features as input for the first-layer mood classifier, which will be described in the next two paragraphs. For the

second-layer mood classifier, We use a sparsity-embeded multi-class logistic regression classifier. We append the outputs of different tasks on the first-layer to the original mood features as input for the second-layer mood classifier.

The features we use for the first-layer mood classifier include features related to the photographing techniques and the features related to the human faces in the movie. The technique related features are listed in the following. **Edge histogram:** a 96-dimension edge histogram feature from Gracenote Vfx 1.0 set. **Color histograms:** The first part of this feature is a 8-dimension gray-level histogram of the image. It is computed from the 64-dimension gray-histogram feature from Gracenote Vfx 1.0 by cumulating every eight bins into one. The second part of this feature is a 4-dimension gray-level histogram and a 4-dimension saturation histogram from Gracenote Vfx 1.0 set. **Hue count:** 1-dimension feature indicating the number of quantized hues that are present in an image. This feature represents the simplicity of the color composition of the image. To compute this feature, we quantize the hue space into 16 bins and compute the hue histogram for the pixels that have saturation value higher than 0.3 and have intensity value between 0.25 - 0.95. We then count the bins whose volume is larger than 1/16 of the largest volume among all bins. **Gray ratio:** 1-dimension feature indicating the ratio of pixels that look achromatic. It is computed as the ratio of pixels that have saturation value lower than 0.3. **Dark ratio:** 1-dimension feature indicating the ratio of pixels that look dark. It is computed as the ratio of pixels that have gray-level intensity lower than 0.25. **Global color statics:** This include a 3-dimension feature vector indicating the mean values of the HSV color channels of the image, and a 3-dimension feature vector indicating the variance for each channel. **Local color statics:** This include a 96-dimension feature vector indicating the mean and variance for all 16 equally separated blcoks

on the image, respectively for the HSV channels. **Optical flow related features:** Based on the OPTFLOW feature in Gracenote Vfx 1.0 feature set, two optical flow related features are used. One is the maximal OPTFLOW value and the other is the mean OPTFLOW value.

The face related features, which are extracted based on detected faces on the frames are listed in the following. The face detector is implemented with the OPENCV library. **Face number:** a 1-dimension feature indicating the name of faces detected in the image. **Mean face position:** a 2-dimension feature vector indicating the mean coordinates of all face centers. **Mean face color:** a 3-dimension feature vector indicating the mean values of the corresponding HSV channels of the face regions. It is set to 0 if no face is detected in the image. **Mean face size:** a 1-dimension feature indicating the mean size of all faces in the image. **Largest face position:** a 2-dimension feature vector indicating the position of the largest face in the image. It is set to zero if no face is detected in the image. **Largest face size:** a 1-dimension feature indicating the relative size of the largest face in the image. **human interaction type:** a 1-dimension feature, indicating whether this image has a single person, or two persons, or a group.

Dataset. Our algorithm allows to use heterogeneous datatsets for different tasks, which is an important advantage for research that aims at utilizing contextual information. It is difficult to have a dataset with different types of labels. In this work, we use three different datasets for the corresponding tasks. For scene categorization, we use the dataset developed by [31] which contains 5 indoor scene categories and 8 outdoor scene categories. Considering the learning burden, we cluster the images into three categories: indoor, natural outdoor, and man-made outdoor, respectively with 930, 1472, and 1457 images. For face

expression classification, we use the JEFFE database [95], consisting of 7 expression categories as introduced earlier. For video mood category, we use the dataset from Gracenote, consisting of 9 mood categories: peaceful, sad, extravagant, easygoing, gentle, thoughtful, dark, coarse, and powerful, with 10 videos in each category. We conduct a 10-fold cross-validation on this set. Each fold contains 9 videos for training and 1 videos for testing for each category.

Evaluation. We measure the performance of the algorithm in two ways. The first method is to evaluate the mood classification accuracy based on frames. Since our algorithm is trained based on individual frames, it is straight-forward to measure the accuracy based on frames for each category, as Equation 6.8.

$$acc(i) = \frac{1}{N_i} \sum_{j=1}^{N_i} \mathbb{I}[\hat{l}(f_{ij}) == l(f_{ij})] \tag{6.8}$$

where f_{ij} is the j^{th} frame in the i^{th} category. $\hat{l}(f_{ij})$ is the estimated label for frame f_{ij} and $l(f_{ij})$ is its corresponding ground-truth label. $\mathbb{I}(\cdot)$ is the indicator function, which equals to 1 when the formula inside stands and 0 otherwise. N_i is the number of frames in the i^{th} category. We then compute the overall accuracy by averaging the frame-based accuracies for each category, as Equation 6.9, where N_c is the number of mood categories.

$$avg\_acc(i) = \frac{1}{N_c} \sum_{i=1}^{N_c} acc(i) \tag{6.9}$$

The second method to evaluate the performance is based on the video clip. In the testing stage, based on the category-scores for each frame of the same video clip, we then make a estimation of the mood for the video clip by some post processing techniques by summing the scores given to each frame of the video and estimate the video mood label as the category with the maximal score, as shown Equation 6.10. v_k is the k^{th} video and f_{kj} is the j^{th} frame of it. c indicates

Table 6.1: Summary of results for baseline method, CCM method, and the proposed FECCM method, for video mood classification.

	Chance	**Base** Model	**2-CCM** (Heitz et.al.[53])	**FECCM** (Our method)
Test Accuracy (%) (frame-based metric)	11.1	28.5	32.6	**36.7**
Test Accuracy (%) (video-based metric)	11.1	32.2	34.4	**40.0**

the candidate category, varying from 1 to N_c.

$$\hat{l}(v_k) = \arg\max_{c \in \{1:N_c\}} \sum_{j=1}^{N_k} score(f_{kj}, c) \tag{6.10}$$

After getting estimation for each video clip, we measure the overall classification accuracy as Equation 6.11.

$$avg\_acc\_video(i) = \frac{1}{N_v} \sum_{i=1}^{N_v} \mathbb{I}[\hat{l}(v_k) == l(v_k)] \tag{6.11}$$

Table 6.1 shows the experiment results, for the base model, CCM model without feedback and our feedback enabled CCM model. For each model, we report both results which are evaluated respectively on the frame level and on the video level. We note that the FE-CCM model outperforms the base model and the CCM model without feedback. The benefits of our model include: 1) the feedback mechanism draws the first layer classifier to focus on important modes, instead of optimizing the intermediate results. 2) the retraining mechanism helps the model to compromise between different database and get less over-fit.

Figure 6.6: Difficult examples of detecting foreground objects in night scenes due to: 1) Dramatic illumination changes; 2) low contrast between foreground and background.

6.2.2 Night Surveillance Video Understanding

Detecting foreground objects for night surveillance videos remains a challenging problem in scene understanding. Though many efforts have been made for robust background subtraction and robust object detection respectively, the complex illumination condition in night scenes makes it hard to solve each of these tasks individually. In practice, we see these two tasks are coupled and can be combined to help each other. Under night scenes, many existing background subtraction methods and object detection methods suffer much from either heavy false alarm due to dramatic lighting changes or missing detection as the foreground color is very closed to the background in local due to low contrast, as shown in Figure 6.6.In this section, we apply the FECCM approach to combine the background subtraction task and the object detection task into a generic framework. We compose the following tasks into the FECCM framework: background subtraction, person detection, car detection, and motorbike detection. We give the implementation details as follows.

Background Subtraction. For the first-layer background subtraction, we build a gaussian model for the gray-scale value of each pixel when being background. For a new frame, it is considered to be a foreground pixel when its difference to the model mean is larger than a threshold T (refer to codes for the exact value).

The model is updated dynamically as follows.

$$\theta^l(p) = (1-\alpha) \times \theta^{l-1}(p) + \alpha \times I^l(p) \text{ if } p \in \text{foreground} \tag{6.12}$$

$$\theta^l(p) = (1-\beta) \times \theta^{l-1}(p) + \beta \times I^l(p) \text{ if } p \in \text{background} \tag{6.13}$$

where p indicates a pixel in the image, $\theta^l(p)$ is the Gaussian mean of the background model for the pixel p at the l^{th} frame, $\theta^{l-1}(p)$ is the Gaussian mean of the background model for the pixel p at the $(l-1)^{th}$ frame, and $I^l(p)$ is the gray-scale value of the pixel p in the l^{th} frame. α and β are the update factors, where β is set to be larger than α (In our implementation, $\alpha = 0.05$ and $\beta = 0.2$).

For the second-layer background subtraction, we use a logistic classifier, which classifies a pixel as foreground or background. The input feature vector includes the original feature input of the first-layer background subtraction (the absolute difference between the pixel value and the Gaussian mean), the binary output of the first-layer background subtraction, and the binary output at the pixel from each of the first-layer object detectors.

Object Detection. For the first-layer object detectors, we use histogram of oriented gradients (HOG) features [18] and apply the deformable-parts-based model in [33]. The deformable-parts-based model contains a mixture of components, allowing for better modeling of the variety of objects within a category. Each component contains a coarse root-filter that serves as a global template for the object, and higher resolution part-filters for different localized parts of the object. In our implementation, we use 8 part-filters. The spatial locations of the object and parts is modeled via a star-graph. The deformable-parts-model is trained discriminatively via a latent SVM. A detailed description of the model can be found in [33]. In our implementation, we first divide the training images into 2 groups based on the time period. For each group, we further divide

the object boxes into 2 sub-groups based on the aspect ratio of the ground-truth bounding boxes. For each of these sub-groups, we generate a left-right flipped version of each image and then use them to train 2 components respectively for the left and right poses of the object. Therefore, we have 8 components in total for an object template.

On the second layer, an object detector is a classifier which re-scores all the candidate boxes detected from the first-layer object detector with an extremely low threshold. The classifier is a RBF-kernel SVM classifier, whose input includes the top-left and bottom-right coordinates $(x1, y1, x2, y2)$ of a candidate box, the first-layer object detector output score for the candidate box, and a score which is the mean value of the first-layer background subtraction outputs of all pixels inside the candidate box.

Dataset: We use the road surveillance dataset and the gate entrance dataset built by Industrial Technology Research Institute for experiments. For each dataset, We use 5 videos for training and 10 videos for testing (3000 – 6000 frames per video) . For each frame, the foreground regions are labeled with bounding boxes and object categories.

Evaluation: We evaluate the object detection performance via the average precision (AP) of precision-recall curves as in [28]. We evaluate the final foreground detection output with the F1 measure, computed as follows.

$$precision = \frac{1}{N} \sum_i \frac{GT^i \cap Y_1^i}{\sum_p Y_1^i(p)} \tag{6.14}$$

$$recall = \frac{1}{N} \sum_i \frac{GT^i \cap Y_1^i}{\sum_p GT^i(p)} \tag{6.15}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{6.16}$$

where GT^i is the ground-truth foreground map for the i^{th} test image, and Y_1^i is

Table 6.2: Performance of background subtraction

	F1 measure
Pixel-based Gaussian Model	0.385
Subspace Method [154]	0.514
Our Method	**0.622**

Table 6.3: Performance of object detection

	AP (Average Precision)		
	Car	Person	Motorbike
Part-based Method [33]	0.556	0.310	0.223
Our Method	**0.610**	**0.352**	**0.425**

the detected foreground map for the i^{th} test image. $GT^i(m,n)$ equals to 1 when the pixel p belongs to foreground, otherwise equals to 0.

Table 6.2 and Table 6.3 give results for background subtraction and object detection respectively. Note that with one single model, our method outperforms a state-of-the-art background subtraction method in [154] and a state-of-the-art object detection method in [33]. Some visualized results from the proposed algorithm are given in Figure 6.7.

6.3 Robotic Applications

In this section, we applied the FE-CCM algorithm onto multiple robotic applications.

Figure 6.7: Examples of the results on surveillance videos in the two ITRI datasets. Each row corresponds to 2 examples. In each example, the left image shows the groundtruth foreground objects (green for "car", blue for "motorbike") and detected objects (red for "car", yellow for "motorbike"), and the right image shows the detected foreground pixels (in pink mask). Best viewed in color.

6.3.1 Robotic Grasping

Given an image and a depthmap (Figure 6.8), the goal of the learning algorithm in a grasping robot is to select a point to grasp the object (this location is called the grasp point, [124]). It turns out that different categories of objects demand different strategies for grasping. In prior work, Saxena et al. [124, 125] did not use object category information for grasping. In this work, we use our FE-CCM to combine object classification and grasping point detection.

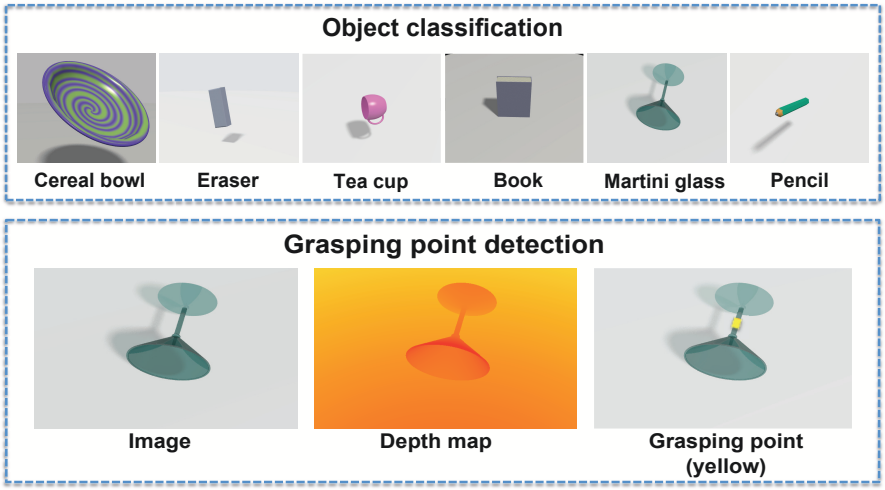


Figure 6.8: Examples in the dataset used for the grasping robot experiments. The two tasks considered were a six-class, object classification task and grasping point detection task.

Implementation: We work with the labeled synthetic dataset by Saxena et al. [124] which spans 6 object categories and also includes an aligned pixel level depth map for each image, as shown in Figure 6.8. The six object categories include spherically symmetric objects such as cerealbowl, rectangular objects such as eraser, martini glass, books, cups and long objects such as pencil.

For grasp point detection, we compute image and depthmap features at each point in the image (using codes given by [124]). The features describe the response of the image and the depth map to a bank of filters (similar to Make3D) while also capturing information from the neighboring grid elements. We then use a regression over the features. The output is a confidence score for each point being a good grasping point. In an image, we pick the point with the highest score as the grasping point.

For object detection, we use a logistic classifier to perform the classification. The output of the classifier is a 6-dimensional vector representing the log-odds

Table 6.4: Summary of results for the the robotic grasping experiment. Our method improves performance in every single task.

Model	Graping point Detection (% accuracy)	Object Classification (% accuracy)
Images in testset	6000	1200
Chance	50	16.7
All features direct	87.7	45.8
Our base-model	87.7	45.8
CCM (Heitz et. al.)	90.5	49.5
FE-CCM	**92.2**	**49.7**

score for each category. The final classification is performed by assigning the image to the category with the highest score.

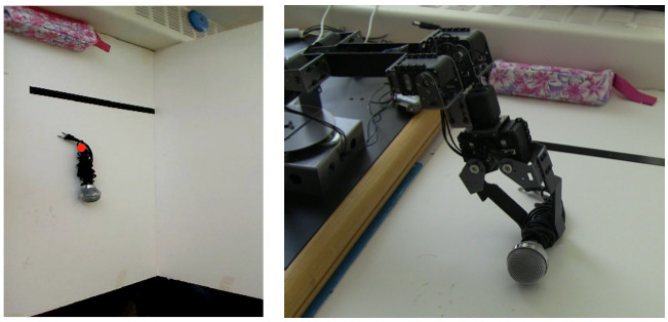


Figure 6.9: Left: the grasping point detected by our algorithm. Right: Our robot grasping an object using our algorithm.

Results: We evaluate our algorithm on a dataset published in [124], and perform cross-validation to evaluate the performance on each task. We use 6000 images for grasping point detection (3000 for training and 3000 for testing) and 1200 images for object classification (600 for training and 600 for testing). Table 6.4 shows the results for our algorithm's ability to predict the grasping point, given an image and the depths observed by the robot using its sensors. We see that our FE-CCM obtains better performance over all-features-direct and CCM (our implementation). Figure 6.9 shows an example of our robot grasping an object.

6.3.2 Object-finding Robot

Given an image, the goal of an object-finding robot is to find a desired object in a cluttered room. As we have discussed earlier, some types of scenes such as living room are more likely to have objects (e.g., shoes) than other types of scenes such as kitchen. Similarly, office scenes are more likely to contain tv-monitors than kitchen scenes. Furthermore, it is also intuitive that shoes are more likely to appear on the supportive surface such as floor, instead of the vertical surface such as the wall. Therefore, in this work, we use our FE-CCM to combine object detection with indoor scene categorization and geometric labeling.

Implementation: For scene categorization, we use the indoor scene subsets in the Cal-Scene Dataset [31] and classify an image into one of the four categories: bedroom, living room, kitchen and office. For geometric labeling, we use the Indoor Layout Data [51] and assign each pixel to one of three geometry classes: ground, wall and ceiling. We use the same features and classifiers for scene categorization as in Section 2.4.

For object detection, we use the PASCAL 2007 Dataset [28] and our own shoe dataset to learn detectors for four object categories: shoe, dining table, tv-monitor, and sofa. We first use the part-based object detection algorithm in [33] to create candidate windows, and then use the same classifiers as described in Section 2.4.

Results: We use this method to build a shoe-finding robot, which is built on the Blue robot of Cornell Personal Robotics Lab, as shown on Figure 6.10-left. With a limited number of training images (86 positive images in our case), it is hard to train a robust shoe detector to find a shoe far away from the camera.

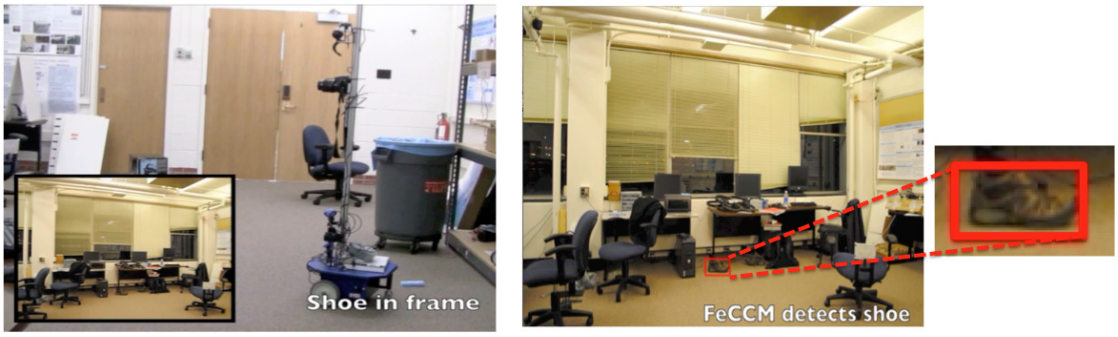


Figure 6.10: Left: Blue robot of Cornell Personal Robotics Lab (used for finding shoes here), which has a camera to take photos of a scene. Right: the shoed detected using our algorithm.

However, using our FE-CCM model, the robot learns to leverage the other tasks and performs more robust shoe detection. Figure 6.10-right shows a successful detection. For more details and videos, please see [89].

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

Incorporating contextual information for enhanced scene understanding tasks has received significant attention in recent works. To completely understand a scene, we want to understand various properties of the scene. Each of these properties correspond to an individual task in scene understanding. Intuitively, all these properties are related, and can serve as contextual attributes between each other. In this thesis, our goal is to leverage the contextual information between tasks for enhanced holistic scene understanding. We considered a two-layer cascade of classifiers, which are repeated instantiations of the original tasks, with the output of the first layer fed into the second layer as input. The first-layer classifiers can be considered as attribute learners, which produce outputs to represent various aspects of the scene. The second-layer target classifiers can then leverage the contextually informative output from various tasks on the first layer. To better optimize the second-layer outputs, we proposed three algorithms, which result in capturing contextual information at multiple levels, ranging from contextual interactions between objects and regions to contextual interactions between different tasks.

First, to leverage contextual information between multiple vision tasks, we propose Feedback Enabled Cascaded Classification Models (FE-CCM), which jointly optimizes all the sub-tasks, while requiring only a 'black-box' interface to the original classifier for each sub-task. The training of our two-layer cascade involves a feedback step that allows later classifiers to provide earlier classifiers information about which error modes to focus on. We show that our method im-

proves performance in *all* the sub-tasks in the domain of scene understanding, where we consider depth estimation, scene categorization, event categorization, object detection, geometric labeling and saliency detection. Besides, we apply the FECCM algorithm on a variety of applicable scenarios, including robotic assistive systems, video applications, and images aesthetics.

Secondly, we consider sharing contextual information between related tasks. We propose the θ-MRF algorithm, which encourages the spatially or semantically related classifiers to share their parameters over the contextual features. We model the independence properties between the parameters for each location and for each task, by defining a Markov Random Field (MRF) over the parameters. In particular, two sets of parameters are encouraged to have similar values if they are spatially close or semantically close. Our method is, in principle, complementary to other ways of capturing context such as the ones that use a graphical model over the labels instead. In extensive evaluation over two different settings, of multi-class object detection and of multiple scene understanding tasks (scene categorization, depth estimation, geometric labeling), our method beats the state-of-the-art methods in all the four tasks.

Third, we discover new contextual attributes from images with only object annotations, to capture the contextual information between objects and regions. Conventional approaches in this direction typically utilize the labeled categories in the images. As a consequence, they inadvertently commit to a fixed scale of interactions. We propose two new contextual cues: (1) Contextual-Meta-Object (CMO), which automatically captures multi- scale interactions between objects and the unlabeled regions; (2) Group-Of-Object (GRP), which automatically captures arbitrary-order interactions between objects that demonstrate consis-

tent spatial, scale, and viewpoint interactions with each other. These contextual patterns are then used as contextual cues for object detection and scene categorization tasks. Our experiments on a variety of datasets show that using these two contextual patterns can boost the performance of object detection and scene categorization.

7.1 Future Work

The following are different directions to be pursued in the future.

7.1.1 Integrating feedback in CCM, θ-MRF, and new attributes

One potential extension is to integrate the proposed algorithms in this thesis. Though the proposed algorithms all aim at improving the performance of the target scene understanding tasks, they have different focuses and may perform complementarily to each other. As initial efforts, we have tried integrating the three mechanism: feedback in CCM, θ-MRF and group attributes, in order to improve the performance of scene categorization. In this experiment, we consider the object detection tasks and scene categorization tasks as attribute learners on the first layer, and scene categorization tasks on the second layer.

We first discover groups of objects as new attributes based on the object annotations in the dataset, and then add corresponding attribute learners for those groups. To incorporate the θ-MRF algorithm, we connect the second-layer classifiers' parameters if they are semantically related. To decide whether two classifiers are related or not, we compute the ratio of co-occurring object categories

in images of those two scene categories. If the two types of scenes share more common objects, the link between the two classifiers is stronger.

We train the model in an EM-style algorithm. In the feed-back stage, we fix the parameters on both layers, and estimate the first-layer preferred outputs as what we do in the feedback step for the FE-CCM model in Chapter 2. In the feed-forward stage, we fix the preferred outputs of both layers, and learn the parameters on both layers. To learn the second-layer parameters, we use the learning algorithm in the θ-MRF model in Chapter 3, taking into account the connections of the second-layer classifiers. The whole learning algorithm can be summarized in Algorithm 4, where $\mathcal{Z}$ and $\mathcal{Y}$ are the sets of preferred outputs of the classifiers on the respective layers, and Θ and Ω are the sets of parameters on the respective layers. The inference algorithm is simply the feed-forward inference approach described in Chapter 2.

Algorithm 4: Learning algorithm of integrating feedback in CCM, θ-MRF, and group attributes

1. Discover groups of objects from the training set with object annotations. Add group attribute learners with the discovered group annotations.

2. Initialize first-layer latent variables $\mathcal{Z}$ with the ground-truth annotations or discovered annotations $\mathcal{Y}$.

2. Do until convergence or maximum iteration: {

Feed-foward step: Fix latent variables $\mathcal{Z}$, estimate the parameters Θ using Equation 2.5 and estimate the parameters Ω using Equation 3.2 and Equation 3.3.2.

Feed-back step: Fix the parameters Θ and Ω, compute latent variables $\mathcal{Z}$ using Equation 2.7.

}

Experiment results. We use the MIT indoor dataset, which contains 15613 images from 67 scene categories and 423 labeled object categories. Since only

2743 images in the dataset have object annotations, we select 15 categories that have more than 50 training images annotated, as listed in Table 5.5-Row 1. We utilize 152 object categories present more than 20 times in images of the 15 scene categories. We discover 52 groups containing 2 to 6 objects. In our CCM framework, we have 152-category object detectors, 52-category group detectors and 15-category scene classifiers on the first layer, and 15-category scene classifiers on the second layer. Each scene classifier is a binary classifier for each scene category. Our learning algorithm takes 5 iterations.

Table 7.1 shows the performance of following methods. **Base method**: The base method compute two types of features: GIST-color features [106] and Spatial Pyramids (SP) features [71], and use the one-vs-all SVM classifiers with RBF kernel. The GIST-color are computed by concatenating the three 320-dimensitional GIST descriptor of the RGB channels of the image. The spatial pyramid features are computed with the implementation provided by [71]. We use a vocabulary of size 200 and three levels in the pyramid. **State-of-the-art (GIST-SP-OBJ-DPM)**: Pandey et al. [107] recently proposed the use of a deformable part-based model (DPM) for a scene categorization, which implicitly captures concurrent regions within a scene. DPM, Combined with the outputs of the base method and the outputs of the object detectors, by multiplying their softmax-transformed outputs, achieved the previous state-of-the-art performance on the same dataset. The other methods listed include the proposed algorithms in previous Chapters and the integrating algorithm in the section. **FE-CCM method**: the method proposed in Chapter 2. **θ-MRF method**: the method proposed in Chapter 3. **CCM method with group attributes**: the method proposed in Chapter 5. **Integrating method**: the proposed method in this section that integrates feedback, θ-MRF, and group attributes. These meth-

Table 7.1: Scene categorization results for the base method, the state-of-the-art method, FE-CCM method, θ-MRF method, CCM with Group-attribute method, and the method integrating the three algorithms.

	Base	**State-of-the-art GIST-SP-OBJ-DPM [107]**	**FE-CCM**	**θ-MRF**	**CCM with Group-Attr**	**Integrating feedback, θ-MRF and groups**
Mean Accuracy (%)	53.1	55.9	56.6	55.4	57.2	**63.2**

ods are built based on the CCM structure. The first-layer object detectors use HOG features, followed by the latent-SVM classification methods. The first-layer scene classifiers use the GIST-color features, followed by the one-vs-all SVM classifiers with RBF kernel. The second-layer scene classifiers take the first-layer outputs as input, and use logistic regression classifiers.

We see that integrating the three methods can further improve each individual method proposed earlier, indicating the complementary property between the methods. In the future, it would be interesting to continue this direction for larger systems with different types of scene understanding tasks.

7.1.2 Mining high-level image patterns

One interesting direction is to continue exploiting intelligent approaches to discover high-level image patterns/modes for the challenging scene understanding tasks such as image aesthetics assessment, image mood classification, and image memorability prediction. So far in this thesis we have only considered discovering the co-occuring composites of objects and regions for object detection and scene recognition tasks. We have previously shown that the structured composites we discover are more powerful than considering only the individ-

ual objects for discriminating the different scene categories. Similarly, if we look at other challenging tasks such as image aesthetics assessment, we realize that it would be difficult to say whether the presence of the saturated color is a favoring or non-favoring element for high aesthetic quality. Instead, the saturated color co-occuring with flowers might be a preferable mode of aesthetics. This observation motivates us that it is important to exploit such underlying patterns for the challenging tasks. Furthermore, we also observe that image properties like aesthetics might not only depend on the appearance of objects and regions, but also other scene attributes such as depth, saliency, and so on. Therefore, it would be interesting to discover patterns that are composed of co-occruing attributes that represent different properties, such as flowers with saturated color, ocean scene with the sun-set color, larger depth at the top region in an outdoor scene, and so on. Besides taking advantage of the group discovery algorithm in this thesis, some other techniques such as sparse coding and deep learning might also be potential directions for the future study.

7.1.3 Extending to scalable systems

We have applied the feedback enabled cascaded classification models (FECCM) onto multiple applications. The results have shown the effectivity of the model in combining related tasks. Since the model allows composing black-box classifiers and does not require considerate knowledge about the individual tasks and their connections, it is suitable for improving the performance of large systems that are performing multiple related tasks simultaneously. In terms of computational cost, an advantage of the model is that the inferences of the classifiers on the same layer can be easily paralleled, which results in a relatively low com-

putation cost in the testing stage. However, due to the EM-style learning mechanism, the computational cost can be high due to the iterations especially when the number of tasks increases and the total number of training data increases correspondingly. Another problem is that, in the current setting, whenever we add new tasks into the composing fr4mework, we need to retrain the model from the beginning. This makes it difficult to scale the model for composing more and more tasks when the corresponding training data and ground-truth labels are available. To solve this, I think one potential direction to investigate can be introducing some incremental learning mechanisms into the learning process, so that less computation is needed for training when new tasks or new data are available.

7.1.4 Applying to interdisciplinary scenarios

So far we only consider visual data as input and vision related applications in this thesis. However, both the FECCM and θ-MRF models are not restricted to visual inputs or vision related tasks. There are many interdisciplinary applications that take various forms of information captured by different sensors as input. For example, in the previously discussed applications such as movie mood classification, we have only analyzed the visual information in the movie. However, there is also plenty of information in the form of audio and text (subtitle) that is highly related to the mood of a movie. One interesting direction would be to apply the FECCM model to compose the tasks in different domains, and investigate how to adjust the model to better incorporate the information from different domains.

Another potential interdisciplinary scenario to be considered is the recently booming social network. Many applications such as user targeted data management and advertisement need to perform user-specific content understanding. It would be an interesting direction to extend the θ-MRF model for such applications, e.g. training user specific classifiers and encouraging the parameter sharing between connected users.

APPENDIX A

RELATED PUBLICATIONS

Books and Journal papers:

- Congcong Li, Tsuhan Chen. Visual Aesthetic Quality Assessment of Digital Images. Chapter in Book: Perceptual Digital Image: Methods and Applications. To be published by CRC Press, in 2012.
- Congcong Li, Adarsh Kowdle, Ashutosh Saxena, Tsuhan Chen. Towards Holistic Scene Understanding: Feedback Enabled Cascaded Classification Models. in IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), July, 2012.
- Congcong Li, Tsuhan Chen. Aesthetic Visual Quality Assessment of Paintings. IEEE Journal of Selected Topics in Signal Processing (IJSTSP), vol.3, no. 2, pp. 236-253, April, 2009.

Conference papers:

- Congcong Li, Adarsh Kowdle, Ashutosh Saxena, Tsuhan Chen. Towards Holistic Scene Understanding: Feedback Enabled Cascaded Classification Models. In Neural Information Processing Systems Conference 2010 (NIPS 2010).
- Congcong Li, Devi Parikh, Tsuhan Chen. Automatic Discovery of Groups of Objects for Scene Understanding. To appear in International Conference on Computer Vision and Pattern Recognition 2012 (CVPR 2012).

- Congcong Li, Devi Parikh, Tsuhan Chen. Exploiting Regions Void of Labels to Extract Adaptive Contextual Cues. To appear in International Conference on Computer Vision 2011 (ICCV 2011).
- Congcong Li, Ashutosh Saxena, Tsuhan Chen. θ-MRF: Capturing Spatial and Semantic Structure in the Parameters for Scene Understanding. In Neural Information Processing Systems Conference 2011 (NIPS 2011).
- Congcong Li, Alexander C. Loui, Tsuhan Chen. "Towards Aesthetics: A Photo Quality Assessment and Photo Selection System." In ACM Multimedia Conference 2010 (ACM MM 2010).
- Congcong Li, Andrew Gallagher, Alexander C. Loui, Tsuhan Chen. Aesthetic Visual Quality Assessment of Consumer Photos with Faces. In IEEE International Conference on Image Processing 2010 (ICIP 2010).
- Congcong Li*, Adarsh Kowdle*, Ashutosh Saxena, Tsuhan Chen. "A generic model to compose vision modules for holistic scene understanding." Workshop on Parts and Attributes, European Conference on Computer Vision, 2010 (ECCV 2010 workshop).
- Congcong Li, TP Wong, Norris Xu, Ashutosh Saxena. "FeCCM for Scene Understanding: Helping the Robot to Learn Multiple Tasks." Video contribution, International Conference on Robotics and Automation 2011 (ICRA 2011 video).
- Congcong Li, Chih-Wei Lin, Shiaw-Shian Yu, Tsuhan Chen. "Joint Optimization of Background Subtraction and Object Detection for Night Surveillance." In IEEE International Conference on Image Processing 2011 (ICIP 2011).

BIBLIOGRAPHY

[1] MSRC 21-class Dataset. http://research.microsoft.com/en-us/projects/objectclassrecognition/.

[2] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned Salient Region Detection. In *CVPR*, 2009.

[3] A. Agarwal and B. Triggs. Monocular human motion capture with a mixture of regressors. In *CVPR Workshop on Vision for HCI*, 2005.

[4] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, December 2005.

[5] S Bengio, F Pereira, and Y Singer. Group sparse coding. In *NIPS*, 2009.

[6] Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards ai. In *Large-Scale Kernel Machines*, 2007.

[7] M.B. Blaschko and C.H. Lampert. Object localization with global and local context kernels. In *BMVC*, 2009.

[8] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[9] G. E. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. John Wiley & Sons, 1992.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge [u.a.]: Univ. Press, 2004.

[11] C. Brezinski, M. Redivo-Zaglia, G. Rodriguez, and S. Seatzu. Multi-parameter regularization techniques for ill-conditioned linear systems. *Mathematics and Statistics*, 94(2):203–228, 2003.

[12] S. Charles Brubaker, Jianxin Wu, Jie Sun, Matthew D. Mullin, and James M. Rehg. On the design of cascades of boosted ensembles for face detection. *IJCV*, 77(1-3):65–86, 2008.

[13] Rich Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

[14] C.D. Cerosaletti and A.C. Loui. Measuring the perceived aesthetic quality of photographic images. In *in Proceedings of First International Workshop on Quality of Multimedia Experience*, pages 282–289, San Diego, CA, USA, July 2009.

[15] Myung Jin Choi, Joseph J. Lim, Antonio Torralba, and Alan S. Willsky. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010.

[16] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, 2008.

[17] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *Comput. Vis. Image Underst.*, 61(1), 1995.

[18] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[19] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Studying aesthetics in photographic images using a computational approach. In *European*

Conference on Computer Vision (ECCV), pages 21–26, Graz, Austria, May 2006.

[20] AP Dawid and S.L. Lauritzen. Hyper markov laws in the statistical analysis of decomposable graphical models. *The Annals of Statistics*, 1993.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J of Royal Stat. Soc., Series B*, 39(1):1–38, 1977.

[22] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009.

[23] Sagnik Dhar, Vicente Ordonez, and Tamara L. Berg. High level describable attributes for predicting aesthetics and interestingness. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 122–133, Colorado Springs, CO, USA, June 2011.

[24] S.K. Divvala, D. Hoiem, J.H. Hays, A.A. Efros, and M. Hebert. An empirical study of context in object detection. In *CVPR*, 2009.

[25] Chuong B. Do, Chuan-Sheng Foo, and Andrew Y. Ng. Efficient multiple hyperparameter learning for log-linear models. In *NIPS*, 2007.

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. Pascal (voc2008). http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[27] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL VOC2006 Results.

[28] Mark Everingham, Luc Gool, Chris Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/.

[29] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[30] Li Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *CVPR*, 2003.

[31] Li. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. *CVPR*, 2005.

[32] P. Felzenszwalb, R. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. http://people.cs.uchicago.edu/ pff/latent-release4/.

[33] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI, IEEE Transactions on*, 32(9):1627 –1645, sep. 2010.

[34] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Discriminatively trained deformable part models, release 3. http://people.cs.uchicago.edu/~pff/latent-release3/.

[35] Rob Fergus, Hector Bernal, Yair Weiss, and Antonio Torralba. Semantic label sharing for learning with many categories. In *ECCV*, 2010.

[36] Rob Fergus, Yair Weiss, and Antonio Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*, 2009.

[37] S. Fidler, M. Boben, and A. Leonardis. Evaluating multiclass learning strategies in a generative hierarchical framework for object detection. In *NIPS*, 2009.

[38] Michael Fink and Pietro Perona. Mutual boosting for contextual inference. In *In NIPS*, 2004.

[39] Y. Freund and R. E. Schapire. Cascaded neural networks based image classifier. In *ICASSP*, 1993.

[40] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *EuroCOLT*, 1995.

[41] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *CVPR*, 2010.

[42] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, 2008.

[43] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. Object categorization using co-occurrence, location and appearance. In *CVPR*, Anchorage, AK, 2008.

[44] Pierre Garrigues and Bruno Olshausen. Group sparse coding with a laplacian scale mixture prior. In *NIPS*, 2010.

[45] M.N. Gibbs and D.J.C. Mackay. Variational gaussian process classifiers. *Neural Networks, IEEE Trans*, 2000.

[46] Ian Goodfellow, Quoc Le, Andrew Saxena, Honglak Lee, and Andrew Ng. Measuring invariances in deep networks. In *NIPS*, 2009.

[47] Stephen Gould, Jim Rodgers, David Cohen, Gal Elidan, and Daphne Koller. Multi-class segmentation with relative location prior. *IJCV*, 80(3), 2008.

[48] K. Grauman and T. Darrell. Unsupervised learning of categories from sets of partially matching image features. In *CVPR*, 2006.

[49] Joo V. Graa, Kuzman Ganchev, Ben Taskar, and Fernando Pereira. Posterior vs. parameter sparsity in latent variable models. In *NIPS*, 2009.

[50] L.K. Hansen and P. Salamon. Neural network ensembles. *PAMI*, 12(10):993–1001, 1990.

[51] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009.

[52] Daniel Heinz. Hyper markov non-parametric processes for mixture modeling and model selection. In *CMU*.

[53] G. Heitz, S. Gould, A. Saxena, and D. Koller. Cascaded classification models: Combining models for holistic scene understanding. In *NIPS*, 2008.

[54] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, 2008.

[55] Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. In *N. Comp*, 2006.

[56] D. Hoiem, A. A. Efros, and M. Hebert. Closing the loop on scene interpretation. In *CVPR*, 2008.

[57] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Putting objects in perspective. *IJCV*, 2008.

[58] Junzhou Huang, Tong Zhang, and Dimitris Metaxas. Learning with structured sparsity. In *ICML*, 2009.

[59] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *NIPS*, 2010.

[60] Wei Jiang, A.C. Loui, and C.D. Cerosaletti. Automatic aesthetic value assessment in photographic images. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 920 –925, Singapore, July 2010.

[61] Yan Ke, Xiaoou Tang, and Feng Jing. The design of high-level features for photo quality assessment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 419–426, New York, NY, USA, June 2006.

[62] Seyoung Kim and Eric P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, 2010.

[63] J. Kittler, M. Hatef, R. P.W. Duin, and J. Matas. On combining classifiers. *PAMI*, 20:226–239, 1998.

[64] Hema Koppula, Abhishek Anand, Thorsten Joachims, and Ashutosh Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *Neural Information Processing Systems (NIPS)*, 2011.

[65] Adarsh Kowdle, Congcong Li, Ashutosh Saxena, and Tsuhan Chen. A generic model to compose vision modules for holistic scene understanding. In *ECCV Workshop on Parts and Attributes*, 2010.

[66] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[67] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *ICCV*, 2005.

[68] S Kumar and Singh. Discriminative fields for modeling spatial dependencies in natural images. In *NIPS*, 2004.

[69] C.H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

[70] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

[71] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[72] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade*. Springer, 1998.

[73] Y. J. Lee and K. Grauman. Foreground focus: Unsupervised learning from partially matching images. In *IJCV*, 2009.

[74] Y.J. Lee and K. Grauman. Object-graphs for context-aware category discovery. In *CVPR*, 2010.

[75] Bastian Leibe, Aleš Leonardis, and Bernt Schiele. Robust object detection with interleaved categorization and segmentation. *Int. J. Comput. Vision*, 77:259–289, May 2008.

[76] C. Li, A. Saxena, and T. Chen. θ-mrf:capturing spatial and semantic structure in the parameters for scene understanding. In *NIPS*, 2011.

[77] C. Li, A. Saxena, and T. Chen. θ-mrf:capturing spatial and semantic structure in the parameters for scene understanding. In *NIPS*, 2011.

[78] Congcong Li, Trista Chen, Peter Dunker, Markus Cremer, and Tsuhan Chen. Composing vision modules for video mood classification. In *Gracenote Inc. Technical Report*, 2010.

[79] Congcong Li and Tsuhan Chen. Aesthetic visual quality assessment of paintings. *Selected Topics in Signal Processing, IEEE Journal of*, 3:236–252, April 2009.

[80] Congcong Li and Tsuhan Chen. *Visual Aesthetic Quality Assessment of Digital Images (Chapter in Book: Perceptual Digital Image: Methods and Applications)*. CRC Press, 2013.

[81] Congcong Li, A. Gallagher, A.C. Loui, and Tsuhan Chen. Aesthetic quality assessment of consumer photos with faces. In *Image Processing (ICIP), IEEE International Conference on*, pages 3221 –3224, Hong Kong, China, September 2010.

[82] Congcong Li, A. Gallagher, A.C. Loui, and Tsuhan Chen. Aesthetic quality assessment of consumer photos with faces. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 2010.

[83] Congcong Li, Adarsh Kowdle, Ashutosh Saxena, and Tsuhan Chen. Feedback enabled cascaded classication models for scene understanding. In *NIPS*, 2010.

[84] Congcong Li, Adash Kowdle, Ashutosh Saxena, and Tsuhan Chen. Towards holistic scene understanding: Feedback enabled cascaded classifi-

cation models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, July 2012.

[85] Congcong Li, Chi-Wei Lin, Shawn-Shian Yu, and Tsuhan Chen. Joint optimization of background subtraction and object detection for night surveillance. In *Image Processing (ICIP), IEEE International Conference on*, 2011.

[86] Congcong Li, Alexander C. Loui, and Tsuhan Chen. Towards aesthetics: a photo quality assessment and photo selection system. In *Proceedings of the international conference on Multimedia*, MM '10, 2010.

[87] Congcong Li, Devi Parikh, and Tsuhan Chen. Exploiting regions void of labels to extract adaptive contextual cues. In *ICCV*, 2011.

[88] Congcong Li, Devi Parikh, and Tsuhan Chen. Automatic discovery of groups of objects for scene understanding. In *CVPR*, 2012.

[89] Congcong Li, TP Wong, Norris Xu, and Ashutosh Saxena. FECCM for scene unverstanding: Helping the robot to learn multiple tasks. In *Video track ICRA. Online URL: http://chenlab.ece.cornell.edu/projects/FECCM/*, 2011.

[90] Li-Jia Li, R. Socher, and Li Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *CVPR*, 2009.

[91] Li-Jia Li, Hao Su, Eric P. Xing, and Li Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *NIPS*, 2010.

[92] L.J. Li and Li Fei-Fei. What, where and who? classifying event by scene and object recognition. In *ICCV*, 2007.

[93] Percy Liang, Francis Bach, Guillaume Bouchard, and Michael I. Jordan. Asymptotically optimal regularization in smooth parametric models. In *NIPS*, 2010.

[94] J.J. Lim, P. Arbel andez, Chunhui Gu, and J. Malik. Context by region ancestry. In *ICCV*, 2009.

[95] Michael J. Lyons, Shigeru Akamatsu, Miyuki Kamachi, and Jiro Gyoba. Coding facial expressions with gabor wavelets. In *Third IEEE International Conference on Automatic Face and Gesture Recognition*, 1998.

[96] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *ECCV*, 2008.

[97] L. Marchesotti, F. Perronnin, D. Larlus, and G. Csurka. Assessing the aesthetic quality of photographs using generic image descriptors. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.

[98] M. Marszalek and C. Schmid. Semantic hierarchies for visual object recognition. In *CVPR*, 2007.

[99] M. Marszalek and C. Schmid. Semantic hierarchies for visual object recognition. In *CVPR*, 2007.

[100] Y. Matsuda. *Color Design (in Japanese)*. Tokyo, Japan: Asakura Shoten, 1995.

[101] Daniel Munoz, J. Bagnell, and Martial Hebert. Stacked hierarchical labeling. In *ECCV*, 2010.

[102] Kevin Murphy, Antonio Torralba, and William T. Freeman. Using the forest to see the trees: A graphical model relating features, objects, and scenes. In *NIPS*, 2003.

[103] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*, 89:355–368, 1998.

[104] Sahand Negahban and Martin J. Wainwright. Joint support recovery under high-dimensional scaling: Benefits and perils of l1,-regularization. In *NIPS*, 2008.

[105] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42:145–175, 2001.

[106] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. In *IJCV*, 2001.

[107] Megha Pandey and Svetlana Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*, 2011.

[108] D. Parikh, C.L. Zitnick, and Tsuhan Chen. From appearance to context-based recognition: Dense labeling in small images. In *CVPR*, 2008.

[109] D. Parikh, C.L. Zitnick, and Tsuhan Chen. Unsupervised learning of hierarchical spatial structures in images. In *CVPR*, 2009.

[110] Dennis Park, Deva Ramanan, and Charless Fowlkes. Multiresolution models for object detection. In *ECCV*, 2010.

[111] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[112] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Conditional random fields for object recognition. In *In NIPS*, 2004.

[113] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *ICCV*, 2007.

[114] N. Rasiwasia and N. Vasconcelos. Holistic context modeling using semantic co-occurrences. In *CVPR*, 2009.

[115] N. Rasiwasia and N. Vasconcelos. Holistic context modeling using semantic co-occurrences. In *CVPR*, 2009.

[116] A Roverato. Hyper inverse wishart distribution for non-decomposable graphs and its application to bayesian inference for gaussian graphical models. *Scandinavian Journal of Statistics*, 2002.

[117] R.Tibshirani. Regression shrinkage and selection via the lasso. *J Royal Stat. Soc. B*, 58(1):267–288, 1996.

[118] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[119] M. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, 2011.

[120] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, 2011.

[121] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, 2011.

[122] A. Saxena, S. Chung, and A. Ng. Learning depth from single monocular images. In *NIPS*, 2005.

[123] A. Saxena, M. Sun, and A.Y. Ng. Make3d: Learning 3d scene structure from a single still image. *PAMI*, 31, 2009.

[124] Ashutosh Saxena, Justin Driemeyer, Justin Kearns, and Andrew Y. Ng. Robotic grasping of novel objects. In *NIPS*, 2006.

[125] Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng. Robotic grasping of novel objects using vision. *IJRR*, 27(2):157–173, 2008.

[126] Ashutosh Saxena, Jamie Schulte, and Andrew Y. Ng. Depth estimation using monocular and stereo cues. In *IJCAI*, 2007.

[127] J. Sivic, B. C. Russell, A. Zisserman, W. T. Freeman, and A. A. Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, 2008.

[128] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005.

[129] E.B. Sudderth, A. Torralba, W.T. Freeman, and A.S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005.

[130] Erik B. Sudderth, Antonio Torralba, William T. Freeman, and Alan S. Willsky. Depth from familiar objects: A hierarchical model for 3d scenes. In *CVPR*, 2006.

[131] C. Sutton and A. McCallum. Joint parsing and semantic role labeling. In *CoNLL*, 2005.

[132] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS*, 2003.

[133] Hanghang Tong, Mingjing Li, Hong jiang Zhang, Jingrui He, and Changshui Zhang. Classification of digital photos taken by photographers or home users. In *In Proceedings of Pacific Rim Conference on Multimedia*, pages 198–205, Tokyo, Japan, November 2004.

[134] A. Torralba, K. Murphy, and W. Freeman. Contextual models for object detection using boosted random fields. In *In NIPS*, 2005.

[135] A. Torralba and A. Oliva. MIT outdoor scene dataset. http://people.csail.mit.edu/torralba/code/spatialenvelope/index.html.

[136] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychol Rev*, 113(4), 2006.

[137] Antonio Torralba. Contextual priming for object detection. *Int. J. Comput. Vision*, 53(2):169–191, 2003.

[138] Antonio Torralba and Aude Oliva. Depth estimation from image structure. *IEEE PAMI*, 24:1226–1238, September 2002.

[139] A. Toshev, B. Taskar, and K. Daniilidis. Object detection via boundary structure segmentation. In *CVPR*, 2010.

[140] I Tsochantaridis, T Hofmann, and T Joachims. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.

[141] Z.W. Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008.

[142] Jasper Uijlings, Koen van de Sande, Arnold Smeulders, Theo Gevers, Nicu Sebe, and Cees Snoek.

The most telling windows for image categorization. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2011/workshop/uva.pdf.

[143] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[144] P. Viola and Michael J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[145] G. Wang and D.A. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *ICCV*, 2009.

[146] Lior Wolf and Stanley Bileschi. A critical view of context. *Int. J. Comput. Vision*, 69:251–261, August 2006.

[147] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. ieee conference on computer vision and pattern recognition. In *CVPR*, 2010.

[148] Bangpeng Yao and Li Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *CVPR*, 2010.

[149] Che-Hua Yeh, Yuan-Chen Ho, Brian A. Barsky, and Ming Ouhyoung. Personalized photograph ranking and selection system. In *Proceedings of the international conference on Multimedia*, MM '10, pages 211–220, Firenze, Italy, October 2010.

[150] Chun-Nam John Yu and Thorsten Joachims. Learning structural svms with latent variables. In *ICML*, 2009.

[151] Wei Yu, A.B. Ashraf, Yao-Jen Chang, Congcong Li, and Tsuhan Chen. 3d

augmented markov random field for object recognition. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 2010.

[152] M. Zeiler, D. Krishnan, G. Taylor, and R. Fergus. Deconvolutional networks. In *CVPR*, 2010.

[153] Yimeng Zhang and Tsuhan Chen. Efficient kernels for identifying unbounded-order spatial features. In *CVPR*, 2009.

[154] Y. Zhao, H. Gong, L. Lin, and Y. Jia. Spatial-temporal patches for night background modeling by subspace learning. In *ICPR*, 2008.